IMPLEMENTASI ALGORITME *IMPROVED PARTICLE SWARM OPTIMIZATION UNTUK* OPTIMASI KOMPOSISI BAHAN MAKANAN UNTUK MEMENUHI KEBUTUHAN GIZI PENDERITA PENYAKIT DIABETES MELITUS

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Gregorius Dhanasatya Pudyakinarya
NIM: 145150207111122

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

Implementasi Algoritme *Improved Particle Swarm Optimization* Untuk Optimasi Komposisi Bahan Makanan Untuk Memenuhi Kebutuhan Gizi Penderita Penyakit Diabetes Melitus

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Gregorius Dhanasatya Pudyakinarya
NIM: 145150207111122

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin, S.Si, M.Kom	Dr.Eng. Fitra A. Bachtiar, S.T, M.Eng
NIK: 201201 850719 1 001	NIK: 201201 840628 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 18 Juli 2018

Gregorius Dhanasatya Pudyakinarya

NIM: 145150207111122

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa yang yang telah melimpahkan berkat, rahmat, dan pertolongan-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Implementasi *Algoritme Improved Particle Swarm Optimization* Untuk Optimasi Komposisi Bahan Makanan Untuk Memenuhi Kebutuhan Gizi Penderita Diabetes Melitus".

Penyusunan laporan skripsi ini dapat terselesaikan dengan bantuan dari beberapa pihak yang turut serta membimbing dan menyertai penulis dalam proses pengerjaanya. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada:

1. Bapak Imam Cholissodin, S.SI, M.Kom dan Bapak Dr.Eng. Fitra Abdurrachman Bachtiar, S.T, M.Eng selaku dosen pembimbing skripsi atas segala waktu, tenaga, dan ilmu yang telah diberikan untuk membimbing dan mengarahkan penulis dalam proses penyusunan skripsi ini.
2. Ibu Fauziatul Firdaus, S.Gz selaku ahli gizi yang bertindak sebagai pakar dalam skripsi ini atas segala waktu, tenaga, ilmu, dan informasi yang diberikan kepada penulis.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D., Bapak Ir. Heru Nurwasito M.Kom, Bapak Suprapto, S.T, M.T, dan Bapak Edy Santoso, S.Si, M.Kom selaku Dekan, Wakil Dekan I, Wakil Dekan II, dan Wakil Dekan III Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D, Bapak Agus Wahyu Widodo, S.T, M.Cs, dan Bapak Muhammad Tanzil Furqon, S.Kom, M.CompSc selaku Ketua Jurusan, Ketua Program Studi dan Sekretaris Program Studi Teknik Informatika.
5. Seluruh dosen dan staff Fakultas Ilmu Komputer Universitas Brawijaya atas segala waktu, tenaga, dan ilmu yang diberikan kepada penulis.
6. Orang tua, kakak, adik, dan seluruh keluarga besar atas segala nasehat, kasih saying, perhatian, dan semua pengorbanan yang telah diberikan kepada penulis.
7. Keluarga besar Marem, IHC, dan Pams yang selalu memberikan motivasi, hiburan, nasehat, dan mendampingi segala kegiatan penulis baik di dalam maupun di luar lingkungan Universitas.
8. Teman-teman Kelas J dan K Teknik Informatika yang membantu penulis selama proses perkuliahan di Fakultas Ilmu Komputer Universitas Brawijaya.
9. Anggota BIOS Filkom, KMK Filkom dan UAKKat Universitas Brawijaya yang telah memberikan wadah bagi penulis dalam berkembang dalam kegiatan non akademik di Universitas Brawijaya.

Penulis menyadari bahwa dalam penyusunan laporan skripsi ini masih banyak kekurangan dan jauh dari sempurna. Oleh karena itu, kritik serta saran yang membangun sangat penulis harapkan. Akhir kata, penulis berharap laporan skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, 18 Juli 2018

Penulis

gregpudya@gmail.com

ABSTRAK

Diabetes Melitus merupakan salah satu penyakit dengan jumlah korban terbanyak di Indonesia. Tingginya penderita diabetes di Indonesia dikarenakan minimnya pengetahuan masyarakat mengenai kendali makanan yang sehat yang mengakibatkan pola makan mereka menjadi buruk. Akibatnya, banyak masyarakat Indonesia yang belum memenuhi keseimbangan asupan gizi yang menjadi bagian terpenting dalam mengatur pola makan yang baik dan sehat. Informasi mengenai pola makan yang tepat diperlukan bagi penderita diabetes untuk memperbaiki kondisi kesehatan mereka. Algoritme *Particle Swarm Optimization* (PSO) seringkali digunakan dalam melakukan sebuah kasus optimasi dengan hasil yang baik dan optimal, terlebih terdapat pengembangan menjadi *Improved Particle Swarm Optimization* (IPSO) yang semakin meningkatkan performa PSO. Oleh karena itu, penelitian ini merancang sebuah sistem optimasi komposisi bahan makanan untuk kebutuhan gizi penderita Diabetes Melitus dengan menggunakan algoritme *Improved*-PSO. Hasil yang didapatkan dari penelitian ini berupa parameter-parameter *Improved*-PSO optimal yaitu jumlah populasi = 150, nilai koefisien akselerasi = 2;1, serta sistem konvergen pada iterasi ke 550. Selain itu, dari hasil analisis global menunjukkan bahwa perhitungan gizi dari sistem dapat memenuhi kebutuhan gizi pasien dengan selisih toleransi ± 10% dari perhitungan pakar.

Kata kunci: optimasi, komposisi bahan makanan, kebutuhan gizi, *Improved Particle Swarm Optimization* (IPSO), Diabetes Melitus.

ABSTRACT

Diabetes Mellitus is one of the diseases with the highest number of casualties in Indonesia. The high diabetics in Indonesia are due to the lack of public knowledge about healthy food controls that result in poor diet. As a result, many people have not met the balance of nutritional intake that is the most important part in managing a good and healthy diet. Information on the right diet is needed for diabetics to improve their health condition. The Particle Swarm Optimization (PSO) algorithm is often used in performing optimization cases with good and optimal results, in particular, there is development to Improved Particle Swarm Optimization (IPSO) which further improves PSO performance. Therefore, this study designs an optimization system for the composition of food ingredients for the nutritional needs of people with Diabetes Mellitus using Improved-PSO algorithm. The results obtained from this study are optimized Improved-PSO parameters that are population number = 150, acceleration coefficient value = 2, 1, and convergent system on iteration to 550. In addition, from the results of global analysis shows that the nutrient calculation of the system can meet the nutritional needs of patients with a difference of tolerance ± 10% of expert calculations.

Keywords: *optimization, food ingredients composition, nutritional needs, Improved Particle Swarm Optimization (IPSO), Diabetes Mellitus.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Diabetes merupakan penyakit yang ditandai dengan sindroma hiperglikemia kronis serta gangguan metabolisme karbohidrat, lemak serta protein yang diakibatkan oleh insufisiensi sekresi insulin maupun aktivitas endogen insulin/keduanya (Suryani, 2015). Diabetes juga merupakan penyakit jangka panjang yang kronis serta ditandai dengan kadar gula atau glukosa yang sangat tinggi dan melebihi batas normal. Glukosa sendiri merupakan komponen penting bagi kesehatan karena merupakan sumber energi utama bagi otak maupun sel-sel yang membentuk otot serta jaringan pada tubuh manusia (Marianti, 2016). Penyakit Diabetes Melitus ini dapat diklasifikasikan menjadi Diabetes Melitus (DM) tipe-1, tipe-2, DM tipe lain dan DM Gestasional.

Berdasarkan data dari *International Diabetes Federation*, sebanyak total 425 juta orang di dunia menderita penyakit Diabetes Melitus dan sebanyak 159 juta kasus ditemukan pada kawasan *West Pasific*, sedangkan Indonesia termasuk satu dari 21 negara yang tergabung dalam teritori *IDF Western Pacific* itu sendiri dengan lebih dari 10 juta kasus penderita penyakit diabetes di Indonesia pada tahun 2017. Hal ini menunjukkan bahwa penderita diabetes di Indonesia terhitung tinggi di dunia. Dan menurut Dirjen Pencegahan dan Pengendalian Penyakit, dr. H. M. Subuh, MPPM (kompas.com, 18/4-2016) menyatakan bahwa sebagian besar penderita diabetes mengunjungi dokter dalam keadaan kronis yang berarti pengetahuan pasien akan kendali makanan yang cenderung terlambat. Hal ini muncul karena kurangnya pengetahuan mengenai pola makan yang sehat dari para penderita penyakit Diabetes Melitus yang mengakibatkan penderita diabetes selalu meningkat tiap tahunnya.

Pengaturan pola makan bagi penderita penyakit diabetes dengan memperhatikan pedoman 3J harus dilakukan, yaitu dengan memperhatikan jenis, jumlah, dan jadwal makan. Contohnya adalah harus tepat dalam menentukan jumlah perhitungan kebutuhan kalori dan zat gizi yang sesuai dengan status gizi penderita (Suryani, 2015). Menurut wawancara dengan seorang Ahli Gizi dari DietIndo bernama Fauziatul Firdaus, S.Gz. menyatakan bahwa pasien telah diberikan arahan mengenai pola makan yang baik namun belum diterapkan dengan baik seperti memakan makanan dengan kandungan lemak yang berlebih atau memakan camilan dengan tingkat karbohidrat tinggi. Seperti yang ditulis oleh Kementrian Kesehatan Republik Indonesia pada tahun 2016, Menteri Kesehatan menyatakan bahwa salah satu faktor predominan diabetes sendiri adalah obesitas. Pola makan yang kaya kalori, garam, lemak jenuh dan gula, dan rendah serat dapat menyebabkan peningkatan berat badan dan meningkatkan resiko diabetes. Keadaan inilah yang menyebabkan cukup tinggi prevalensinya pada masyarakat Indonesia yang berarti sebagian besar masyarakat Indonesia menerapkan pola makan yang salah. Penanganan pola makan serta komposisi makanan harus sesuai dengan keadaan penderita agar dapat mengurangi kadar

gula dalam darah. Sehingga, hal tersebut menjadi salah satu cara penanganan yang paling efektif bagi penderita penyakit diabetes. Informasi mengenai pola makan yang tepat sangat diperlukan bagi seluruh penderita diabetes baik dalam tingkat awal terlebih pada pasien tingkat kronis.

Istikomah (2017) telah menggunakan algoritme PSO untuk melakukan optimasi pemenuhan gizi yang menghasilkan rekomendasi komposisi makanan selama 7 hari. Penelitian yang dilakukan oleh Eliantara, Cholissodin, dan Indriati (2016) dalam melakukan optimasi pemenuhan gizi keluarga juga menggunakan algoritme PSO dengan hasil sistem dapat menghemat biaya sebesar 39,31%. Lalu dalam penelitan yang menerapkan algoritme *Improved*-PSO untuk optimasi komposisi bahan pakan ayam petelur oleh Hasjilda, Cholissodin, dan Widodo (2017) telah mampu memberikan hasil dengan biaya 50,41% lebih murah dibandingkan dengan harga yang didapatkan dari peternak. Algoritme *Improved*-PSO tersebut terbukti dapat meringankan biaya yang dikeluarkan dalam membeli bahan makanan sekaligus memberikan makanan hasil rekomendasi yang paling optimal.

Dengan adanya permasalahan mengenai pola makan penderita penyakit Sebelumnya, terdapat penelitian yang dilakukan oleh Maryamah (2017) untuk mengoptimasi komposisi makanan pada penderita diabetes melitus dan komplikasinya dengan menggunakan Algoritme Genetika dengan nilai *fitness* 0.01857. Namun, menurut Yonghe et al. (2015) algoritme PSO lebih mudah untuk diaplikasikan dan lebih cepat dalam menemukan titik konvergensi serta penggunaan parameter *constriction factor* dan *inertia weight* secara asinkron yang dapat memaksimalkan kinerja PSO. Oleh karena itu, algoritme IPSO dipilih dalam penyelesaian masalah optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita Diabetes Melitus.

Diabetes Melitus di atas dan mengacu pada penelitian-penelitian sebelumnya, maka penulis akan membuat sebuah sistem cerdas yang tujuannya adalah memberikan informasi mengenai komposisi bahan makanan yang optimal untuk memenuhi kebutuhan gizi para penderita Diabetes Melitus. Pasien Diabetes Melitus yang dijadikan responden merupakan pasien yang menderita Diabetes Melitus tanpa komplikasi dengan rentang usia di atas 40 tahun. Sistem dirancang dengan menggunakan algoritme *Improved*-PSO sebagai metode yang digunakan dalam penyelesaian masalah.

1.2 Rumusan Masalah

Berdasarkan pada latar belakang di atas maka dapat diambil beberapa rumusan masalah, yaitu:

1. Bagaimana implementasi algoritme *Improved Particle Swarm Optimization* dengan parameter yang optimal untuk menghitung optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita penyakit Diabetes Melitus?

2. Bagaimana hasil evaluasi pada penerapan algoritme *Improved Particle Swarm Optimization* untuk menghitung optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita penyakit Diabetes Melitus?

1.3 Tujuan

1. Mengimplementasikan algoritme *Improved Particle Swarm Optimization* dengan parameter yang optimal untuk menghitung optimasi komposisi bahan makanan untuk memenuhi gizi penderita penyakit Diabetes Melitus.
2. Menganalisis hasil evaluasi pada penerapan algoritme *Improved Particle Swarm Optimization* untuk menghitung optimasi komposisi bahan makanan untuk memenuhi gizi penderita penyakit Diabetes Melitus.

1.4 Manfaat

1. Bagi Klinik/Rumah Sakit

   Penelitian ini memberikan bantuan layanan bagi Klinik atau Rumah Sakit dalam menangani penderita penyakit Diabetes Melitus sekaligus dapat memberikan informasi tambahan berupa komposisi bahan makanan yang optimal bagi pasien.

2. Bagi Penderita Penyakit Diabetes Melitus

   Penelitian ini dapat membantu penderita penyakit Diabetes Melitus dalam mendapatkan informasi mengenai komposisi bahan makanan yang optimal dan dapat memenuhi kebutuhan gizinya.

3. Bagi Penulis

   Penulis dapat menerapkan ilmu yang didapatkan sehingga dapat mengimplementasikan algoritme *Improved Particle Swarm Optimization* dalam memperoleh hasil optimal dari komposisi bahan makanan untuk memenuhi gizi penderita penyakit Diabetes Melitus serta menganalisis hasil evaluasi dari penelitian yang dilakukan.

1.5 Batasan Masalah

1. Data komposisi bahan makanan yang digunakan berdasarkan sumber dari Ahli Gizi bernama Fauziatul Firdaus, S,Gz dari Diet Indo.
2. Rumus perhitungan kebutuhan gizi penderita penyakit Diabetes Melitus berdasarkan rumus dari PERKENI (Perkumpulan Endokrinologi Indonesia) dan hasil wawancara dengan Ahli Gizi bernama Fauziatul Firdaus, S.Gz dari Diet Indo.
3. Menggunakan 5 jenis parameter *Particle Swarm Optimization* dan 2 parameter tambahan sebagai pengembangan algoritme *Improved Particle Swarm Optimization*.
4. Rekomendasi ditujukan bagi pasien penderita Diabetes Melitus Tipe 2 tanpa komplikasi dengan rentang usia di atas 40 tahun.

5. Hasil solusi sebatas optimasi bahan makanan dengan tidak mengatur menu dan belanja pasien serta tidak memperhatikan rasa dan harga bahan makanan. Optimasi dilakukan hanya untuk memenuhi kebutuhan gizi pasien.
6. Program diet yang dirancang dengan membuat menu diet maksimal 3 hari atau 3 kombinasi menu dalam rentang waktu satu minggu.

1.6 Sistematika Pembahasan

Penyusunan penelitian ini memiliki sistematika penulisan serta pembahasan dengan rincian sebagai berikut:

BAB 1 Pendahuluan

Pada bab ini berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, hingga sistematika mengenai implementasi algoritme *Improved Particle Swarm Optimization* untuk optimasi kebutuhan makanan bagi penderita penyakit Diabetes Melitus.

BAB 2 Landasan Kepustakaan

Pada bab ini berisi pembahasan mengenai kajian pustaka dan dasar teori yang menunjang dalam penelitian seperti pengertian mengenai penyakit Diabetes Melitus, dasar teori mengenai *Swarm Intellgience*, *Particle Swarm Optimization*, dan *Improved Particle Swarm Optimization*.

BAB 3 Metodologi

Pada bab ini berisi pembahasan mengenai tahapan metodologi penelitian yang berisi studi pustaka, proses pengumpulan data, proses analisis kebutuhan sistem, proses perancangan hingga implementasi sistem, dan proses pengujian dan analisis sistem yang dilanjutkan dengan penarikan kesimpulan dan saran.

BAB 4 Perancangan

Pada bab ini berisi pembahasan mengenai analisa kebutuhan sistem yang digunakan sebagai acuan dalam melakukan perancangan sistem yang menghitung optimasi kebutuhan makanan bagi penderita penyakit Diabetes Melitus dengan menggunakan algoritme *Improved-PSO*.

BAB 5 Implementasi

Pada bab ini berisi pembahasan mengenai implementasi sistem yang dibuat berdasarkan hasil analisa kebutuhan dan perancangan sistem.

BAB 6 Pengujian dan Analisis

Pada bab ini berisi pembahasan mengenai alur pengujian serta analisis terhadap sistem yang telah dibangun.

BAB 7 Penutup

Pada bab ini berisi pembahasan mengenai kesimpulan serta saran penelitian yang diperoleh dari pembuatan dan pengujian sistem.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Studi pustaka yang digunakan dalam penelitian ini berupa hasil-hasil penelitian sebelumnya yang memiliki keterkaitan dengan implementasi algoritme *Particle Swarm Optimization* dalam optimalisasi suatu kebutuhan gizi ataupun komposisi bahan makanan. Penelitian terdahulu yang penulis ambil sebagai sumber kajian pustaka adalah berupa paper ataupun jurnal penelitian. Studi pustaka berupa kajian pustaka ini digunakan dengan tujuan agar semakin memahami permasalahan yang dihadapi sekaligus mencari solusi terbaik yang digunakan dalam penyelesaian masalah. Daftar kajian pustaka yang memiliki keterkaitan dengan penelitian dijelaskan pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka

Judul	Objek	Metode	Hasil
Improved Particle Swarm Optimization Algorithm And Its Application In Text Feature Selection Yonghe, Minghui, Zeyuan, & Lichao (2015)	Berbagai algoritme Improved-PSO sebagai fitur dalam penyeleksian teks	Improved Particle Swarm Optimization	Algoritme PSO yang menggunakan *constriction factor* memiliki hasil yang lebih optimal dibandingkan dengan PSO biasa. Lalu PSO dengan memanfaatkan *inertia weight* sekaligus *constriction factor* secara asinkron meningkatkan hasil dari klasifikasi teks
Implementasi Algoritme Particle Swarm Optimization (PSO) untuk Optimasi Pemenuhan Kebutuhan Gizi Balita Istikomah, Cholissodin, dan Marji (2017)	Komposisi makanan balita	Particle Swarm Optimization	Nilai parameter optimal pada pengujian sebelumnya dijadikan acuan dalam tahap analisis global terhadap 10 data balita yang diambil secara acak. Rata-rata yang pengeluaran bagi orangtua hasil rekomendasi sistem sebesar Rp.25,003.6 dan

Judul	Objek	Metode	Hasil
			selama 7 hari adalah Rp.175,025.44
Optimasi Pemenuhan Kebutuhan Gizi Keluarga Menggunakan Particle Swarm Optimization Eliantara, Cholissodin, dan Indriati (2016)	Komposisi bahan makanan untuk asupan gizi anggota keluarga	Particle Swarm Optimization	Parameter terbaik hasil pengujian sebelumnya digunakan pada data keluarga aktual. Hasilnya berupa rata-rata selisih kebutuhan gizi aktual keluarga. Batas toleransi selisih kebutuhan gizi yang dikonsumsi adalah 10% dan sistem dianggap sudah memenuhi standar pakar
Optimasi Komposisi Pakan Untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum Menggunakan Improved Particle Swarm Optimization (Improved-PSO) Hasjidla, Cholissodin, dan Widodo (2018)	Komposisi makanan ayam petelur	Improved Particle Swarm Optimization	Hasil akhir didapatkan dari parameter Improved-PSO terbaik hasil pengujian dan menghasilkan biaya 50,41% lebih murah dibandingkan dnegan harga yang didapatkan dari peternak telur dengan selisih Rp. 313,68
Implementasi Algoritme Improved Particle Swarm Optimization Untuk Optimasi Komposisi Bahan Makanan Untuk Memenuhi Kebutuhan Gizi Penderita Penyakit Diabetes Melitus (Usulan)	Komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita Diabetes Melitus	Improved Particle Swarm Optimization	Rekomendasi bahan makanan bagi penderita penyakit Diabetes Melitus yang terdiri dari makan pagi, makan siang, dan makan malam

Penelitian mengenai *Improved Particle Swarm Optimization Algorithm and Its Application In Text Feature Selection*, dijabarkan mengenai berbagai macam pengembangan lebih lanjut terhadap algoritme PSO (Yonghe, et al., 2015). Di dalamnya termasuk penggunaan fungsi *constriction factor* dan *inertia weight* yang terbukti dengan penggunaan kedua parameter tersebut menciptakan proses klasifikasi teks menjadi lebih optimal. Karakteristik konvergensi dari PSO juga membuat kombinasi kedua parameter tersebut lebih baik dilakukan secara asinkron. Pengujian yang dilakukan menggunakan algoritme KNN untuk menguji masing-masing parameter *Improved*-PSO dan hanya dilakukan untuk efektifitas seleksi fitur.

Pada penelitian sebelumnya yang dilakukan oleh Istikomah, Cholissodin, dan Marji (2017) tentang Optimasi Pemenuhan Kebutuhan Gizi Balita, dikemukakan bahwa semakin banyak jumlah partikel maka semakin tinggi pula nilai *fitness* dan semakin lama waktu komputasi yang dihabiskan. Selain itu dikemukakan juga bahwa nilai koefisien dianggap dapat mempengaruhi nilai *fitness* yang sangat tinggi. Pada penelitian ini disimpulkan bahwa Implementasi algoritme PSO untuk optimasi Pemenuhan Kebutuhan Gizi Balita dapat dilakukan dengan cara inisialisasi partikel awal secara *random*, setelah mendapatkan nilai partikel dapat melakukan konversi nilai partikel kedalam indeks bahan makanan, hasil konversi digunakan untuk mengetahui nama bahan makanan, berat dan harga. Setelah tahap-tahap tersebut maka dapat dihitung *fitness*, *Update* Kecepatan, *Update* Posisi, *Update PBest* dan *GBest*. *GBest* pada iterasi terakhir dijadikan sebagai Hasil Optimasi.

Pada penelitian sebelumnya yang dilakukan oleh Eliantara, Cholissodin, dan Indriati (2016) mengenai Optimasi Pemenuhan Kebutuhan Gizi Keluarga Menggunakan *Particle Swarm Optimization* yang diterapkan dalam penyelesaian optimasi dilakukan sesuai dengan aturan-aturan umum yang ada dalam algoritme PSO. Representasi partikel berupa angka *random* yang kemudian dinormalisasi untuk menentukan indeks bahan makanan dan menjadi faktor dalam menghitung nilai *fitness.* Keluaran yang dihasilkan berupa susunan bahan makanan untuk jangka waktu selama 7 hari dengan frekuensi makan 3 kali dalam sehari. Hasil yang muncul dari sistem dapat menghemat biaya sebesar 39,31% dan variasi makanan yang beragam.

Pada penelitian sebelumnya yang dilakukan oleh Hasjidla, Cholissodin, dan Widodo (2018) mengenai Optimasi Komposisi Pakan Untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum. Algoritme *Improved*-PSO pada permasalahan optimasi komposisi pakan ayam petelur dapat dilihat dari besarnya nilai *fitness*. Partikel dengan nilai *fitness* tertinggi merupakan partikel yang memiliki solusi paling optimal. Dikatakan optimal jika nilai *penalty* yang dihasilkan mendekati atau sama dengan 0 dan biaya yang dihasilkan juga rendah. Pada penelitian ini disimpulkan bahwa algoritme *Improved*-PSO dapat diterapkan pada Optimasi Komposisi Pakan Untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum.

Berdasarkan dengan penelitian yang telah dilakukan sebelumnya, maka penulis akan menggunakan algoritme *Improved Particle Swarm Optimization*. Algoritme ini sering digunakan dalam perhitungan optimasi komposisi bahan makanan ataupun kebutuhan gizi dengan keluaran yang dihasilkan memiliki selisih yang dihasilkan cukup signifikan apabila dibandingkan dengan *Particle Swarm Optimization* biasa. Sistem akan menghitung kebutuhan nutrisi bagi penderita penyakit Diabetes Melitus dan memberikan hasil berupa komposisi bahan makanan harian dalam beberapa hari.

2.2 Diabetes Melitus

Diabetes Melitus merupakan penyakit multifaktorial, yang ditandai dengan adanya sindroma hiperglikemia kronis dan gangguan metabolisme karbohidrat, lemak, dan protein yang disebabkan oleh insufisiensi sekresi insulin maupun aktivitas endogen insulin atau keduanya (Suryani, 2015). Penyakit ini dibedakan menjadi dua, yaitu Diabetes Melitus tipe 1 dan tipe 2. Pada diabetes tipe 1 terjadi akibat pankreas yang tidak bisa memproduksi insulin, sedangkan pada tipe 2 terjadi jika tubuhnya masih dapat memproduksi insulin, tapi jumlahnya tidak cukup (Suryani, 2015).

Kadar gula darah yang tinggi di dalam tubuh disebabkan oleh tidak sempurnanya proses metabolisme zat makanan dalam sel tubuh (Misdarina dan Ariani, 2012). Kadar gula darah dapat dipantau dengan 4 pilar penatalaksanaan diabetes yaitu edukasi, perencanaan makanan, latihan jasmani, dan terapi obat yang sangat penting karena kadar gula darah merupakan indikator dalam diagnosa penyakit Diabetes Melitus (Misdarina dan Ariani, 2012).

Pengaturan pola makan harus diketahui dan dilaksanakan oleh penderita melitus yaitu perhitungan yang tepat mengenai kebutuhan kalori dan zat gizi yang sesuai dengan status gizi penderita, bukan berdasarkan tinggi rendahnya gula darah (Suryani, 2015). Perencanaan makan untuk pasien bertujuan untuk mencapai serta mempertahankan kadar glukosa darah dan lemak darah normal dengan memperhatikan jenis, jumlah, dan jadwal makan (Suryani, 2015).

Menurut Trijayanto (2016) berikut merupakan faktor resiko seseorang terkena Diabetes Melitus:

1. Riwayat keluarga merupakan faktor risiko utama seseorang mengalami Diabetes Melitus, secara genetic pasien Diabetes Melitus mempengaruhi keturunannya.
2. Usia lanjut pada umumnya merupakan penderita Diabetes Melitus tipe 2.
3. Obesitas yang merupakan keadaan abnormal atau akumulasi lemak berlebihan dan merupakan faktor resiko penyebab terjadinya penyakit degenerative seperti Diabetes Melitus, jantung coroner, dan hipertensi.
4. Jenis kelamin.

5. Kurang olahraga, karena olahraga berperan secara efektif dalam mengontrol Diabetes Melitus seperti melakukan senam khusus Diabetes Melitus tipe 2, berjalan kaki, bersepeda, dll.
6. Gaya hidup yang tidak sehat.
7. Pola makan yang berlebihan dan melebihi kadar kebutuhan kalori juga dapat memicu timbulnya Diabetes Melitus.
8. Merokok.
9. Stres yang dapat meningkatkan risiko diabetes karena produksi hormon kortisol secara berlebihan.
10. Hipertensi yang berhubungan dengan resistensi insulin dan abnormalitas pada sistem renin-angiotensin dan konsekuensi metabolik yang meningkatkan morbiditas.
11. Diet pada penderita Diabetes Melitus yang meliputi pengaturan kalori, dan pemberian asupan karbohidrat, lemak, dan protein dalam ketujuh kelompok penggolongan makanan.

2.3 Kebutuhan Kalori Penderita Diabetes Melitus

Setiap manusia memiliki kebutuhan kalori berbeda-beda tergantung pada usia, jenis kelamin, aktivitas, kondisi fisik, dan lain-lain. Berikut merupakan perhitungan untuk menentukan kebutuhan gizi seseorang yang dalam keadaan normal dan dalam berbagai macam keadaan sakit menurut penelitian yang dilakukan oleh Simamora (2017):

1. Perhitungan untuk menghitung total energi seperti pada Persamaan 2.1.

$$Keb.Energi = AMB * Faktor\ Aktivitas * Faktor\ Stres \quad (2.1)$$

2. Angka Metabolisme Basal didapatkan dari Persamaan 2.2 dan 2.3.

   Laki-Laki

$$AMB = 66 + (13{,}7 * BB) + (5 * TB) - (6{,}8 * U) \quad (2.2)$$

   Perempuan

$$AMB = 66{,}5 + (9{,}7 * BB) + (1{,}8 * TB) - (4{,}7 * U) \quad (2.3)$$

   Keterangan:

   BB = Berat Badan dalam satuan kg

   TB = Tinggi Badan dalam satuan cm

   U = Umur dalam satuan tahun

3. Faktor Aktivitas berisi nilai-nilai faktor yang berbeda untuk tiap jenis aktivitas mulai dari istirahat hingga aktivitas berat yang dilakukan oleh orang tersebut. Begitu pula dengan faktor stres yang berisi nilai faktor yang berbeda sesuai dengan jenis trauma/stres yang dialami pasien mulai dari tidak mengalami stres hingga stres tingkat sangat berat.

4. Kebutuhan total protein yang dibutuhkan sama dengan orang normal yaitu dalam kisaran 10 hingga 15% dari kebutuhan total energi yang dibutuhkan.
5. Kebutuhan total protein yang dibutuhkan sama dengan orang normal yaitu dalam kisaran 10 hingga 25% dari kebutuhan total energi yang dibutuhkan.
6. Kebutuhan total protein yang dibutuhkan sama dengan orang normal yaitu dalam kisaran 60 hingga 75% dari kebutuhan total energi yang dibutuhkan.

Bagi penderita penyakit Diabetes Melitus, terdapat perbedaan dalam menghitung kebutuhan kalori per harinya. Menurut wawancara dengan Ahli Gizi bernama Ibu Fauziahtul Firdaus, S.Gz serta berdasarkan Buku Pengelolaan dan Pencegahan Diabetes Melitus Tipe 2 di Indonesia tahun 2015 oleh Perkumpulan Endokrinologi Indonesia (PERKENI), ada beberapa cara dalam menentukan jumlah kalori yang dibutuhkan para pasien diabetes yaitu sebagai berikut:

1. Perhitungan berat badan ideal (BBI) dapat dilakukan dengan 2 metode:
   - Perhitungan BBI dengan menggunakan rumus Broca dapat dilihat pada Persamaan 2.4.

     $$BBI = 90\% * (TB\ dalam\ cm - 100) * 1\ kg \quad (2.4)$$

     Khusus untuk pria dengan tinggi badan di bawah 160 cm dan wanita di bawah 150 cm, terdapat perubahan rumus seperti pada Persamaan 2.5.

     $$BBI = (TB\ dalam\ cm - 100) * 1\ kg \quad (2.5)$$

     Berdasarkan pada rumus di atas, maka hasilnya dapat dipisah menjadi 3 kategori berat badan yang berbeda, yaitu:
     - Berat Badan Normal: *BBI* ± 10%
     - Berat Badan Kurus: *BBI* - 10%
     - Berat Badan Gemuk: *BBI* + 10%
   - Perhitungan berat badan ideal (BBI) menurut Indeks Massa Tubuh (IMT) dapat dilihat pada Persamaan 2.6.

     $$IMT = BB(kg) / TB * TB(m^2) \quad (2.6)$$

     Berdasarkan rumus di atas, terdapat 3 klasifikasi IMT, yaitu:
     - Berat Badan Kurang < 18,5
     - Berat Badan Normal 18,5 - 22,9
     - Berat Badan Lebih ≥ 23,0
       - Dengan risiko 23,0 - 24,9
       - Obes I 25,0 - 29,9
       - Obes II ≥ 30

2. Setelah mendapatkan nilai berat badan ideal, selanjutnya adalah menghitung Kebutuhan Kalori Basal (KKB). KKB dibagi menurut faktor-faktor yang menentukan kebutuhan kalori antara lain:

   - Jenis Kelamin

     Perhitungan KKB berdasarkan jenis kelamin terdapat dalam Persamaan 2.7 dan 2.8 berikut:

     $$KKB\ Pria = 30\ kalori * BB \quad (2.7)$$

     $$KKB\ Perempuan = 25\ kalori * BB \quad (2.8)$$

   - Umur

     Perhitungan KKB berdasarkan usia dibagi dalam 3 rentang usia yang terdapat pada Persamaan 2.9, 2.10, dan 2.11 di mana kebutuhan kalori dikurangi untuk setiap dekade.

     $$Usia\ 40 - 59\ tahun = 5\% * KKB\ Jenis\ Kelamin \quad (2.9)$$

     $$Usia\ 60 - 69\ tahun = 10\% * KKB\ Jenis\ Kelamin \quad (2.10)$$

     $$Usia\ > 70\ tahun = 20\% * KKB\ Jenis\ Kelamin \quad (2.11)$$

   - Aktivitas Fisik atau Pekerjaan

     Kebutuhan kalori ditambahkan sesuai dengan aktivitas fisik atau pekerjaan pasien. Terdapat 5 kategori aktivitas yang dapat dilihat pada Tabel 2.2 dan perhitungan KKB dijelaskan pada Persamaan 2.12 - 2.17.

     $$Bed\ Rest = 5\% * KKB\ Jenis\ Kelamin \quad (2.12)$$

     $$Keadaan\ Istirahat = 10\% * KKB\ Jenis\ Kelamin \quad (2.13)$$

     $$Aktivitas\ Ringan = 20\% * KKB\ Jenis\ Kelamin \quad (2.14)$$

     $$Aktivitas\ Sedang = 30\% * KKB\ Jenis\ Kelamin \quad (2.15)$$

     $$Aktivitas\ Berat = 40\% * KKB\ Jenis\ Kelamin \quad (2.16)$$

     $$Aktivitas\ Sangat\ Berat = 50\% * KKB\ Jenis\ Kelamin \quad (2.17)$$

Tabel 2.2 Kategori Aktivitas

Kategori Aktivitas	Contoh Profesi
Aktivitas Ringan	Pegawai kantor, guru, ibu rumah tangga
Aktivitas Sedang	Pegawai industri ringan, militer, mahasiswa
Aktivitas Berat	Petani, buruh, militer dalam keadaan latihan
Aktivitas Sangat Berat	Atlet, tukang gali

(Sumber: Pengelolaan dan Pencegahan DM Tipe 2, PERKENI 2015)

- Berat Badan (Status Gizi)

  Pada perhitungan kebutuhan kalori berdasarkan berat badan dibagi dalam 7 rumus yang terdapat pada Persamaan 2.18-2.24.

  $$Status\ Gizi = \frac{BB}{BBI} * 100\% \quad (2.18)$$

  $$(Status\ Gizi = 60 - 70\%) = 30\% * KKB\ Jenis\ Kelamin \quad (2.19)$$

  $$(Status\ Gizi = 70 - 80\%) = 20\% * KKB\ Jenis\ Kelamin \quad (2.20)$$

  $$(Status\ Gizi = 80 - 90\%) = 10\% * KKB\ Jenis\ Kelamin \quad (2.21)$$

  $$(Status\ Gizi = 120 - 130\%) = -(10\% * KKB\ Jenis\ Kelamin) \quad (2.22)$$

  $$(Status\ Gizi = 130 - 140\%) = -(20\% * KKB\ Jenis\ Kelamin) \quad (2.23)$$

  $$(Status\ Gizi\ \geq 140\%) = -(30\% * KKB\ Jenis\ Kelamin) \quad (2.24)$$

3. Tahap terakhir dalam perhitungan kebutuhan kalori harian seorang pasien diabetes adalah dengan mengakumulasi keseluruhan jumlah KKB yang tertera pada Persamaan 2.25, 2.26, 2.27, dan 2.28.

   $$Total\ Kalori = KKB\ Jenis\ Kelamin - KKB\ Umur + KKB\ Aktivitas\ Fisik + KKB\ Berat\ Badan \quad (2.25)$$

   $$Karbohidrat\ (Kkal) = (65\% * Total\ Kalori)/4 \quad (2.26)$$

   $$Protein\ (KKal) = (15\% * Total\ Kalori)/4 \quad (2.27)$$

   $$Lemak\ (KKal) = (20\% * Total\ Kalori)/9 \quad (2.28)$$

2.4 Swarm Intelligence

Pada ilmu komputer dikenal dengan sistem kecerdasan berkoloni binatang. Beberapa sifat alamiah seperti berkoloni (berpasang-pasang), mencari makanan, regenerasi, kecerdasan alami (sosial & personal) dan lainnya, semua itu adalah merupakan hal-hal yang dapat diambil untuk suatu sistem cerdas sintesis dari alam. Menurut Cholissodin dan Riyandani (2016), sistem cerdas yang dimaksud di sini adalah *Swarm Intellligence*, berhubungan dengan sistem alami dan buatan di mana terdiri dari banyak individu atau bisa disebut dengan populasi yang berkoordinasi dengan menggunakan cara kontrol *self-organized* (kecerdasan personal/ terorganisir secara mandiri) dan desentralisasi (kecerdasan sosial dalam berkelompok). Pengertian singkatnya *Swarm Intellligence* adalah Kecerdasan Berkelompok.

Dalam *Swarm Intellligence*, terdapat macam-macam tipe yang masing-masing memiliki keunikannya sendiri-sendiri dan meniru perilaku dari hewan-hewan tertentu, seperti koloni burung (*Particle Swarm Optimization*), semut (*Ant Colony Optimization*), dan lebah (*Bee Colony Optimization*). Contohnya dalam ACO, algoritme ini didapatkan dari cara hidup semut dalam mencari makanan di mana mengandalkan *pheromone* untuk menemukan rute terpendek dari sarangnya ke sumber makanan.

Berikut merupakan beberapa macam karakteristik *Swarm Intellligence* (Cholissodin dan Riyandani, 2016):

- Harus membuat design ulang representasi solusi yang sesuai untuk setiap goal atau *contraint* maupun kasus berbeda.
- Terkadang tidak mendapat suatu hasil yang optimal atau mengalami kegagalan (terjebak pada local optimal, konvergensi dini).
- Beberapa mekanisme alami belum dapat dipahami dengan baik.
- Setiap problem optimasi yang diselesaikan harus sudah terdefinisi dengan baik, mulai dari *input* datanya apa saja, bagaimana representasi solusinya, *contraints*nya apa saja, bagaimana rancangan rumus *fitness*-nya atau *cost*-nya.

Representasi solusi merupakan vektor atau matrik sederhana maupun komplek dalam bentuk tertentu dan spesifik. Biasanya representasi solusi bentuknya berbeda atau dapat berubah secara dinamis sesuai dengan cara pandang seseorang terkait pemahamannya terhadap problem yang diselesaikan.

Dalam uraian lain representasi solusi bisa disebut juga sebagai bentuk partikel atau individu dalam suatu populasi (kumpulan dari banyak individu) yang memiliki nilai posisi tertentu. Nilai-nilai dalam posisi tersebut menyatakan rangkaian solusi yang dibawa oleh masing-masing individu. Sebagai kandidat solusi optimal jika sudah mencapai kondisi konvergen.

a. Inti konsep terkait dengan pemecahan permasalahan-permasalahan yang sulit sekali untuk dimodelkan dalam bentuk matematika kompleks, dan jika dibuat model persamaan matematikanya membutuhkan waktu yang lama.
b. Membutuhkan peranan algoritme *meta-heuristics*.
c. Termasuk *nature-inspired meta-heuristics*, yaitu merupakan suatu *framework* algoritme yang diinspirasi dari alam di mana memiliki kemampuan di mana di dalamnya terdapat banyak sekali teknik untuk sebuah penyelesaian yang optimal ketika dilakukan pencarian atau penelusuran pada berbagai macam permasalahan yang berbeda-beda (hanya dengan sedikit modifikasi).

Berikut merupakan beberapa macam prinsip Kerja *Swarm Intellligence* (Cholissodin dan Riyandani, 2016):

- *Organizing Principles Swarm Intelligence*, berikut adalah fitur utamanya :
  a) *Autonomy*: individu di dalam suatu sistem dapat bekerja secara mandiri (autonom/*self-organized*), di mana dalam mengendalikan perilaku mereka sendiri baik pada tingkat detektor dan efektor.
  b) *Adaptability*: interaksi yang terjadi antar individu dapat timbul melalui komunikasi langsung (*direct*) atau tidak langsung (*indirect*).
  c) *Scalability*: mampu untuk men-*generate* suatu kelompok-kelompok solusi yang ada di mana terdiri dari beberapa, ribuan individu, atau bahkan lebih dengan arsitektur kontrol yang sama.

d) *Flexibility*: di mana tidak ada individu tunggal yang paling utama di dalam suatu populasi tertentu, sehingga pada setiap individu mendapatkan perlakuan yang sama satu sama lainnya, yaitu dapat ditambahkan secara dinamis, dihapus, ataupun dapat diganti.

e) *Robustness*: tidak ada suatu koordinasi yang terpusat secara pasti pada suatu titik solusi (individu) tertentu, sehingga hal ini memungkinkan tidak adanya satu individu pun yang dianggap mengalami suatu kegagalan, hal ini dikarenakan secara desentralisasi telah terbantu oleh individu lainnya.

f) *Massively Parallel*: di mana tugas yang dilakukan atau yang dikerjakan oleh masing-masing individu di dalam populasi adalah sama untuk satu sama lainnya.

g) *Self-organization*: suatu kecerdasan sistem dapat tumbuh tidak hanya bertumpu pada suatu individu tertentu, melainkan dapat pula bertumpu dari keseluruhan individu dalam suatu kelompok atau kawanan.

- Berikut merupakan penjelasan singkat mengenai pengertian *Direct* dan *Indirect Communication* (Cholissodin dan Riyandani, 2016):

  a) *Direct Communication* merupakan suatu komunikasi yang secara eksplisit terjadi pada antar individu yang ada. Berikut adalah contoh-contoh dari *direct communication*:
     - *Update GBest*
     - *Update PBest*
     - *Update* Posisi Partikel
     - *Update* Kecepatan Partikel
     - *Waggle Dance* yang terdapat di dalam *Algoritme Bee Colony*

  b) *Indirect Communication* merupakan suatu komunikasi implisit yang terjadi di antara individu melalui lingkungan yang ada di sekitarnya. Dikenal pula dengan komunikasi *Stigmergy* yang terdiri dari kata Stigma dan Ergon yang berarti bahwa percept yang diterima oleh individu dari lingkungan ketika individu tersebut telah memberikan aksi tertentu terhadap lingkungannya.

2.5 Particle Swarm Optimization

Russell Eberhart (*Electrical Engineer*) dan James Kennedy (*Social-Psychologist*) merupakan penemu Algoritme *Particle Swarm Optimization* (PSO) pada tahun 1995 yang mana merupakan salah satu cabang "*Swarm Intellligence*" yang berdasar kepada algoritme Meta-Heuristik (Cholissodin dan Riyandani, 2016). Algoritme PSO ini merupakan algoritme yang terinspirasi dari perilaku sekawanan burung ketika melakukan komunikasi dan kerja sama antar satu sama lainnya. Sehingga membentuk suatu pola global yang kompleks yang berdasar kepada kecerdasan yang muncul dari perilaku berkomunikasi dan bekerja sama yang dilakukan oleh burung-burung.

Dalam PSO sendiri, setiap individu yang ada di dalam *swarm* atau kelompok disebut dengan partikel di mana berperilaku sebagai seorang agen pada lingkungan yang terdesentralisasi dan bersifat cerdas. Pada setiap partikel di dalam *swarm* sangat berkontribusi pada lingkungan sekitarnya untuk dapat mengikuti pola-pola sederhana agar dapat bekerja sama dan berkomunikasi antar partikel dalam satu *swarm*.

Perilaku kolektif global yang cukup kompleks muncul di dalam *swarm* di mana diadopsi untuk dapat menyelesaikan konsep permasalahan optimasi yang cukup kompleks. Dengan tingkat desentralisasi yang tinggi, kerja sama antar partikel, dan implementasi yang sederhana membuat PSO dapat digunakan dengan optimal untuk menyelesaikan suatu permasalahan optimasi secara efisien.

Pada algoritme *Particle Swarm Optimization* ini terdeapat beberapa proses dalam pencarian solusi sesuai dengan partikel atau ukuran dimensi tertentu:

1. Inisialisasi
2. *Update* kecepatan
3. *Update* posisi dan hitung *fitness*
4. *Update PBest* dan *GBest*

2.6 Improved Particle Swarm Optimization

Algoritme *Improved Particle Swarm Optimization* merupakan sebuah pengembangan dari algoritme *Particle Swarm Optimization* pada umumnya. Terdapat 2 penambahan parameter sebelum dilakukan inisialisasi partikel awal, yaitu *inertia weight* dan *constriction factor*. Penerapan *inertia weight* (*w*) dan *constriction factor* (*K*) dilakukan dalam waktu yang berbeda (asinkron). Nilai (*K*) dan (*w*) tadi dapat diformulasikan menjadi Persamaan 2.29 dan 2.30 (Yonghe, et al., 2016).

$$K = \frac{\cos\left(\frac{2\pi}{Tmax} x \left(\frac{t - Tmax}{2}\right)\right) + 2{,}428571}{4} \tag{2.29}$$

Keterangan:

K = *constriction factor*

Tmax = iterasi maksimal

t = iterasi pada saat itu

$$w = \begin{cases} 0{,}857143 + \left((1 - 0{,}857143) x \left(1 - \frac{t}{T_{max}}\right)\right), GBest_j \neq X_{ij} \\ 0{,}857143 \quad , GBest_j = X_{ij} \end{cases} \tag{2.30}$$

Keterangan:

w = *inertia weight* (bobot inersia)

Tmax = iterasi maksimal

t = iterasi pada saat itu

$GBest_j$ = posisi terbaik partikel ke-*j*

X_{ij} = posisi partikel ke-*i* dimensi ke-*j*

Sebelum melakukan *update* kecepatan, lebih dahulu dilakukan perhitungan nilai C_1 dan C_2. Nilai C_1 dan C_2 merupakan koefisien akselerasi untuk keseimbangan yang lebih baik antara global eksplorasi dan lokal eksploitasi (Imam & Efi, 2016). Konsep ini kemudian disebut dengan *Time Varying Acceleration Coefficients* (TVAC). Global eksplorasi yang dimaksud di sini adalah *fitness* terbaik yang telah dicapai oleh semua partikel dan lokal eksploitasi adalah *fitness* terbaik yang dicapai partikel saat iterasi tertentu. Nilai *range* C_1 dan C_2 yang digunakan adalah [2.5, 0.5] dan [0.5, 2.5] karena terbukti optimal (Ratnaweera, 2004). Berikut merupakan persamaan TVAC pada Persamaan 2.31 (Chen, et al., 2011).

$$C_1 = \left(c_{1f} - c_{1i}\right)\frac{t}{t_{max}} + c_{1i} \quad C_2 = \left(c_{2f} - c_{2i}\right)\frac{t}{t_{max}} + c_{2i} \tag{2.31}$$

Keterangan:

$c_{1f} - c_{1i}\ dan\ c_{2f} - c_{2i}$ merupakan nilai inisial koefisien akselerasi dan merupakan konstanta.

Inertia weight dan *constriction factor* merupakan parameter yang memiliki perbedaan karakteristik, oleh karena itu kedua parameter ini digunakan dalam waktu yang berbeda atau asinkron. *Inertia weight* digunakan untuk menyeimbangkan penelusuran lokal serta global pada setengah awal iterasi. Dilanjutkan dengan parameter *constriction factor* pada sisa iterasi yang berfungsi untuk memastikan bahwa konvergensi mencapai titik yang paling optimal (Yonghe, et al., 2015) dan pada Persamaan 2.32 merupakan rumus untuk melakukan *update* kecepatan pada algoritme *Improved*-PSO.

$$v_{ij} = \begin{cases} w \times V_{ij} + C_1 \times r_1\left(PBest_j - X_{ij}\right) + C_2 \times r_2\left(GBest_j - X_{ij}\right), t < \frac{T_{max}}{2} \\ K[0{,}7V_{ij} + C_1 \times r_1\left(PBest_j - X_{ij}\right) + C_2 \times r_2\left(GBest_j - X_{ij}\right)], t \geq \frac{T_{max}}{2} \end{cases} \tag{2.32}$$

Keterangan:

V_{ij} = kecepatan partikel ke-*i* dimensi ke-*j*

w = *inertia weight*

K = *constriction factor*

Tmax = iterasi maksimal

t = iterasi pada saat itu

$PBest_j$ = posisi terbaik partikel ke-*j*

$GBest_j$ = posisi terbaik partikel dalam *swarm* ke-*j*

X_{ij} = posisi partikel ke-*i* dimensi ke-*j*

r_1 = nilai *random* 1 (nilai acak antara 0-1, dibuat konstan hanya untuk perhitungan manual)

r_2 = nilai *random* 2 (nilai acak antara 0-1, dibuat konstan hanya untuk perhitungan manual)

Tahap-tahap dalam algoritme *Improved*-PSO dalam mencari solusi optimal adalah sebagai berikut:

1. Inisialisasi awal

   Inisialisasi awal partikel PSO biasanya dilakukan dengan menggunakan pengkodean biner yang bertujuan untuk menyederhanakan masalah. Namun, kelemahan pengkodean biner adalah tidak bias menjangkau beberapa titik solusi jika range solusi berada dalam daerah kontinyu (Imam&Efi, 2016). Di sisi lain, pada optimasi fungsi yang kompleks dan memiliki banyak generasi, optimasi biner ke desimal dan sangat membutuhkan banyak waktu. Sehingga, pada inisialisasi partikel awal ini menggunakan metode *Real-Code* PSO (RCPSO). Berikut rumus untuk inisialisasi awal partikel yang dibangkitkan secara random dapat dilihat pada Persamaan 2.33.

   $$x_{ij} = x_{ijmin} + rand[0,1] * (x_{ijmax} - x_{ijmin}) \quad (2.33)$$

   Keterangan:

   x_i = partikel ke-*i*

   x_{ijmin} = batas bawah partikel ke-*i*

   x_{ijmax} = batas atas partikel ke-*i*

   rand [0,1] = batas atas angka permutasi

   Evaluasi partikel merupakan langkah selanjutnya setelah melakukan inisialisasi partikel. Pada tahap ini dilakukan perbandingan nilai *fitness* yang digunakan untuk menentukan *GBest* dan *PBest*. Pada tahap perhitungan nilai *fitness*, diperlukan nilai penalti dan harga dari bahan makanan. Perhitungan nilai penalti dan harga ini diawali dengan menghitung berat bahan makanan yang ditunjukkan pada Persamaan 2.34.

   $$totalberat_{x,y,z} = \left(berat_x * takaranporsi_y\right) * a_z \quad (2.34)$$

   Keterangan:

   $totalberat_{x,y,z}$ = total berat bahan makanan ke-*x*, takaran porsi ke-*y*, waktu makan ke-*z*

   $berat_x$ = berat awal bahan makanan ke-*x*

   $takaranporsi_y$ = takaran porsi bahan makanan jenis-*y*

   a_z = persentase waktu makan

   (pagi=25%, siang=25%, malam=25%, camilan (masing-masing)=10%)

Setelah menghitung berat bahan makanan yang dibutuhkan, dilanjutkan dengan menghitung kandungan gizi masing-masing bahan makanan dengan menggunakan Persamaan 2.35 berikut:

$$nilai\ gizi_{x,y} = \frac{totalberat_x}{berat_x} * gizi_y \quad (2.35)$$

Keterangan:

nilai gizi$_{x,y}$ = nilai gizi berupa kandungan gizi *y* bahan makanan ke-*x*

totalberat$_x$ = total berat bahan makanan ke-*x*

berat$_x$ = berat awal bahan makanan ke-*x*

gizi$_y$ = kandungan gizi *y*

Setelah mendapatkan nilai gizi masing-masing bahan makanan, dilanjutkan dengan mengihtung total jumlah nilai gizi dan rata-rata gizi seperti pada Persamaan 2.36 dan 2.37.

$$jumlahGizi_x = \sum gizi_x \quad (2.36)$$

Keterangan:

jumlahGizi$_x$ = total nilai gizi *x* dalam satu partikel

gizi$_x$ = nilai gizi *x*

$$rataGizi_x = \frac{\sum jumlahGizi_x}{\sum hari} \quad (2.37)$$

Keterangan:

rataGizi$_x$ = rata-rata nilai gizi *x*

jumlahGizi$_x$ = total nilai gizi *x* dalam satu partikel

hari = jumlah hari dalam satu partikel

Pada penelitian yang dilakukan oleh Maryamah (2017), terdapat perhitungan penalti gizi sebelum menentukan nilai *fitness* yang digunakan seperti pada Persamaan 2.38.

$$Total\ Penalti = penalti_1 + penalti_2 + penalti_3 + penalti_4 \quad (2.38)$$

Keterangan:

penalti$_1$ = penalti kalori

penalti$_2$ = penalti karbohidrat

penalti$_3$ = penalti protein

penalti$_4$ = penalti lemak

Penalti merupakan suatu keadaan di mana terdapat nilai yang tidak sesuai aturan. Beberapa aturan yang harus diterapkan dalam proses optimasi makanan bagi penderita diabetes adalah kandungan kalori, karbohidrat, protein, dan lemak yang tidak kurang atau lebih dari kandungan gizi yang

seimbang (Maryamah, 2017). Menurut wawancara dengan Ahli Gizi Fauziatul Firdaus, S.Gz. berikut merupakan perhitungan penalti gizi bagi penderita Diabetes Melitus tanpa komplikasi seperti pada Persamaan (2.39-2.42).

$$Penalti\ Kalori = |(Anjuran\ atau\ kebutuhan\ asupan\ kalori\ pasien - jumlah\ total\ kalori\ yang\ masuk\ dari\ makanan)| \quad (2.39)$$

$$Penalti\ Karbohidrat = |(Anjuran\ atau\ kebutuhan\ asupan\ karbohidrat\ pasien - jumlah\ total\ karbohidrat\ yang\ masuk\ dari\ makanan)| \quad (2.40)$$

$$Penalti\ Protein = |(Anjuran\ atau\ kebutuhan\ asupan\ protein\ pasien - jumlah\ total\ protein\ yang\ masuk\ dari\ makanan)| \quad (2.41)$$

$$Penalti\ Lemak = |(Anjuran\ atau\ kebutuhan\ asupan\ lemak\ pasien - jumlah\ total\ lemak\ yang\ masuk\ dari\ makanan)| \quad (2.42)$$

Setelah mendapatkan nilai penalti gizi, dilanjutkan dengan menghitung variasi bahan makanan dengan menjumlahkan nama bahan makanan yang berbeda dalam satu partikel.

Berikut merupakan rumus perhitungan *fitness* menurut Eliantara (2016) yang digunakan setelah mendapatkan nilai penalti dan variasi pada Persamaan (2.43).

$$Fitness = \frac{1}{Total\ Penalti} * const1 + variasi \quad (2.43)$$

Keterangan:

const1 = penyeimbang nilai *fitness* dikarenakan perhitungan variasi menghasilkan angka puluhan, nilai *const1* adalah 100000

Setelah melakukan evaluasi partikel dengan memanfaatkan perhitungan nilai *fitness*, selanjutnya adalah menentukan nilai *PBest* awal (*Population Best*) dan *GBest* awal (*Global Best*). Nilai *GBest awal* didapatkan dengan memilih 1 *PBest* dengan nilai *fitness* tertingi yang merepresentasikan nilai optimal dalam suatu populasi dalam 1 kali iterasi.

2. *Update* Kecepatan

Setelah mendapatkan nilai *GBest* awal, selanjutnya masuk pada proses iterasi pertama dengan melakukan *update* kecepatan untuk memperbaharui nilai partikel awal dengan menggunakan rumus pada Persamaan 2.32.

3. *Update* Posisi

Setelah mendapatkan nilai kecepatan yang baru, maka dilakukan perhitungan posisi untuk perubahan posisi partikel yang terdapat pada Persamaan 2.44 berikut:

$$X_i^{j+1} = X_i^j + V_i^{j+1} \quad (2.44)$$

Keterangan:

X_i^{j+1} = posisi partikel ke-*i* pada iterasi ke-(*j*+1)

X_i^j = posisi partikel ke-*i* pada iterasi ke-*k*

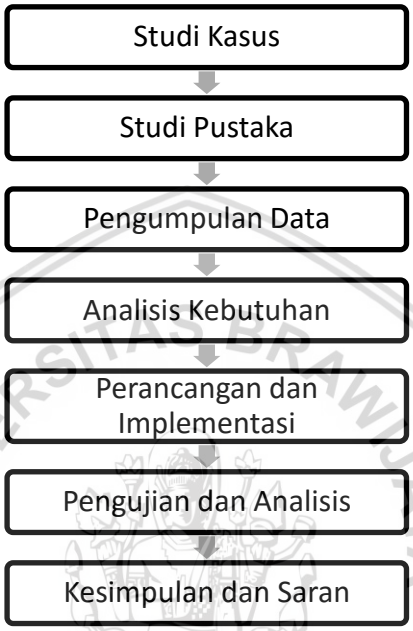
V_i^{j+1} = posisi partikel ke-*i* pada iterasi ke-(*j*+1)

4. *Update PBest* dan *GBest*

Setelah melakukan *update* posisi, langkah selanjutnya adalah proses *update PBest* di mana membandingkan nilai *PBest* sebelum dan sesudah iterasi. Jika nilai *fitness* partikel baru lebih besar dari *fitness PBest* sebelumnya, maka partikel tersebut dijadikan *PBest* terbaru. *Update GBest* dilakukan dengan memilih partikel terbaik dari seluruh anggota *swarm* yang didapatkan dari nilai *fitness PBest* tertinggi dan melanjutkan ke iterasi selanjutnya apabila *stopping condition* belum terpenuhi.

BAB 3 METODOLOGI

Dalam bab ini dibahas mengenai metodologi penelitian yang digunakan dalam Implementasi Algoritme *Improved Particle Swarm Optimization* Untuk Optimasi Komposisi Bahan Makanan Untuk Memenuhi Kebutuhan Gizi Penderita Penyakit Diabetes Melitus. Berikut merupakan gambaran umum alur penyelesaian masalah seperti pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

Berdasarkan pada gambar di atas, dari permasalahan yang muncul dari kondisi pasien penyakit Diabetes Melitus diproses dengan menggunakan algoritme *Improved*-PSO untuk menghasilkan sebuah keluaran berupa optimasi kebutuhan bahan makanan optimal demi memenuhi kebutuhan gizi pasien tersebut. Dari hasil yang dikeluarkan nantinya juga dilakukan evaluasi sistem untuk mengetahui kualitas solusi yang dihasilkan oleh algoritme *Improved*-PSO.

3.1 Tipe Penelitian

Tipe penelitian yang digunakan oleh penulis adalah penelitian non-implementatif analitik. Nonimplementatif berarti penulis melakukan sebuah proses investigasi terhadap suatu keadaan tertentu maupun melakukan analisis terhadap permasalahan yang muncul yang kemudian digunakan sebagai produk utama penelitian. Teknik yang digunakan pleh penulis dalam menghasilkan produk tersebut berupa hasil dari studi kasus serta proses wawancara terhadap objek penelitian yang digunakan. Sedangkan, untuk penelitian analitik merupakan jenis kegiatan penelitian nonimplementatif di mana penulis dalam penelitian ini memberikan sebuah hasil analisis yang dihasilkan dari penjelasan dari hubungan antar elemen yang terdapat pada objek penelitian dengan kondisi tertentu yang diteliti. Tipe penelitian ini secara garis besar merupakan sebuah proses yang

menitikberatkan dalam penggalian informasi dari fenomena tertentu yang akhirnya mampu menjawab pertanyaan-pertanyaan yang muncul pada awal penelitian.

3.2 Strategi Penelitian

Penelitian dimulai dari pengumpulan studi kasus seputar penyakit Diabetes Melitus dan pola makanan pasiennya. Studi kasus ini didapatkan dari buku, jurnal, paper, maupun referensi lainnya dengan tidak lupa untuk melibatkan seorang pakar terkait. Hasil studi kasus tadi digunakan sebagai permasalahan utama dalam penelitian ini yang dilanjutkan pada proses pembelajaran studi pustaka dalam mencari algoritme atau metode dalam menyelesaikan permasalahan tersebut. Pencarian algoritme berdasarkan pustaka dalam penelitian-penelitian sebelumnya yang terdapat pada jurnal penelitian yang telah ada. Hasil pembelajaran dari studi pustaka ini selanjutnya diaplikasikan kedalam sebuah sistem cerdas untuk menyelesaikan permasalahan yang diangkat dan diakhiri dengan proses evaluasi sistem untuk menguji kualitas penelitian.

3.3 Subjek Penelitian

Subjek penelitian merupakan pasien penderita Diabetes Melitus sebagai subjek utama penelitian. Selain itu, terdapat Ahli Gizi yang berperan sebagai pakar dalam bidang kesehatan terutama mengenai penanganan penyakit Diabetes Melitus sebagai sumber informasi utama penelitian sekaligus menjaga agar kualitas penelitian baik dan tetap sesuai dengan kaidah yang sudah ada.

3.4 Lokasi Penelitian

Lokasi penelitian dalam pengumpulan data yang utama adalah di kantor utama Diet Indo sebagai penyedia data dan pelaksanaan penelitian sebagian besar dilakukan di lingkungan Fakultas Ilmu Komputer Universitas Brawijaya.

3.5 Teknik Pengumpulan Data

Pengumpulan data dimulai sepanjang proses penelitian. Data yang pertama berupa daftar bahan penukar makanan yang didapatkan dari Ahli Gizi bernama Fauziatul Firdaus, S.Gz dari Diet Indo, data kedua adalah data 10 pasien penderita penyakit Diabetes Melitus tanpa komplikasi dengan rentang usia di atas 40 tahun dengan kondisi rawat jalan serta dibagi dalam 2 kualifikasi kondisi. Data pasien tersebut didapatkan saat pengembangan sistem cerdas dari sumber data yang sama. Selain kedua data tersebut, dilakukan juga proses wawancara dengan pakar mengenai informasi seputar penyakit Diabetes Melitus dan cara penanganannya dalam bentuk mengatur pola makan yang benar bagi penderitanya.

3.6 Perancangan dan Implementasi

Perancangan sistem dalam optimasi kebutuhan makanan ini dilakukan dalam beberapa tahap yang dimulai dari proses memasukkan data dari pengguna sistem

lalu penentuan parameter *Improved*-PSO awal, inisialisasi partikel awal yang pada iterasi ke-0 langsung kepada perhitungan *fitness*, *PBest* awal, *GBest* awal. Kemudian dilanjutkan pada iterasi berikutnya dan memasuki proses *update* kecepatan, posisi, *fitness*, *PBest*, *GBest* hingga mencapai iterasi maksimal yang ditentukan hungga pada hasil komposisi makanan optimal.

a. Masukan

   Masukan dari pengguna berupa nama, berat badan, tinggi badan, jenis kelamin, usia, dan pekerjaan pasien serta jumlah populasi dan iterasi maksimum.

b. Proses

   Penentuan parameter *Improved*-PSO dilakukan pada awal proses perhitungan sebelum masuk pada inisialisasi partikel awal. Setelah parameter *Improved*-PSO ditentukan, maka dilakukan proses inisialisasi partikel awal dan masuk kepada iterasi ke-0. Pada iterasi ke-0 ini langsung dilakukan perhitungan *fitness*, dilanjutkan dengan menentukan *PBest* dan *GBest* awal. Iterasi kedua berlangsung dan masuk kepada proses *update* kecepatan, posisi, menghitung *fitness*, menentukan *PBest* dan *GBest* lalu masuk pada iterasi selanjutnya hingga mencapai batas iterasi (iterasi maksimal).

c. Keluaran

   Setelah mencapai batas iterasi, maka hasil yang dimunculkan berupa komposisi bahan makanan optimal yang ditampilkan berupa nama makanan, total kalori, dan total harga.

Implementasi sistem menggunakan bahasa pemrograman *PHP* yang dimulai pada proses *input* data dari pengguna hingga hasil yang dikeluarkan. Tahapan pada proses implementasi ini dibagi menjadi 2, yaitu:

a. Implementasi antarmuka, menentukan parameter *Improved*-PSO, proses inisialisasi awal, proses perhitungan pada iterasi ke-0, proses perhitungan pada iterasi ke-1 hingga batas maksimal iterasi hingga menampilkan hasil optimasi menggunakan perangkat lunak *sublime text* dengan bahasa pemrograman *PHP*.

b. Implementasi *database* dengan menggunakan aplikasi *HeidiSQL*.

3.7 Teknik Analisis Data

3.7.1 Pengujian Parameter

Pengujian batas ruang pencarian dimensi dilakukan untuk mendapatkan nilai parameter *Improved*-PSO yang paling optimal. Proses pengujian dari ruang pencarian dimensi ini diakukan sebanyak 10 kali. Pengujian ini dilakukan untuk setiap dimensi. Berikut detail parameter yang digunakan:

a. Jumlah partikel (*popSize*)

b. Koefisien Akselerasi

Pada masing-masing pengujian batas ruang pencarian dimensi, hasil pengujian dari masing-masing parameter secara berurutan digunakan sebagai acuan dalam menghitung parameter yang selanjutnya. Contoh: dari hasil pengujian jumlah partikel menghasilkan bahwa *popSize* dengan jumlah 50 menjadi nilai yang paling optimal dalam menghasilkan nilai *fitness*, maka dalam pengujian selanjutnya, yaitu pengujian koefisien akselerasi yang berupa nilai C_1 dan C_2 parameter *popSize* yang dipakai adalah 50 populasi.

3.7.2 Pengujian Konvergensi

Pada pengujian konvergensi ini dilakukan sebanyak 5 kali dengan menggunakan nilai parameter yang terbaik pada pengujian parameter yang sebelumnya dilakukan. Pengujian sistem dapat dikatakan konvergen apabila dari hasil yang keluar menunjukkan keragaman populasi yang semakin minim disertai dengan nilai *fitness* yang tinggi dan stabil.

3.7.3 Analisis Global

Analisis global merupakan proses terakhir dalam tahap pengujian dan analisis sistem ini. Analisis global ini dilakukan untuk mengetahui kualitas kerja sistem yang dibangun dengan cara membandingkan hasil keluaran sistem hasil optimasi dengan data manual dari Ahli Gizi yang bersangkutan. Pada pengujian ini menggunakan nilai parameter yang terbaik hasil pengujian sebelumnya.

3.8 Kesimpulan dan Saran

Proses penarikan kesimpulan dilakukan setelah proses pengujian dan analisis telah dilakukan serta melakukan evaluasi terhadap hasil yang ada. Penarikan kesimpulan dilakukan dengan tujuan menjawab rumusan masalah dalam penelitian. Setelah penarikan kesimpulan selesai, tahap terakhir yaitu saran-saran dari penulis yang terkait dengan hasil penelitian guna memberikan tambahan informasi sebagai bahan pertimbangan dalam pengembangan penelitian kedepannya.

BAB 4 PERANCANGAN

Bab ini membahas mengenai keseluruhan proses perancangan penelitian mulai dari deskripsi formulasi permasalahan, data yang digunakan, tahapan dan perhitungan penyelesaian masalah dengan menggunakan algoritme *Improved Particle Swarm Optimization*, perancangan pengujian, dan perancangan antarmuka.

4.1 Formulasi Permasalahan

Komposisi makanan bagi penderita penyakit Diabetes Melitus ini merupakan permasalahan yang diselesaikan dalam penelitian ini demi memenuhi gizi pasien Diabetes Melitus. Demi mendapatkan hasil kompoisi yang optimal dengan menggunakan algoritme *Improved Particle Swarm Optimization*, maka terdapat beberapa proses yang digunakan dalam proses perhitungan.

Proses yang pertama merupakan memasukkan data pribadi pasien yang berisi nama, usia, jenis kelamin, berat badan, tinggi badan, dan pekerjaan yang digunakan dalam menentukan kebutuhan kalori harian. Tahap pertama dalam perhitungan kebutuhan kalori harian adalah menghitung Berat Badan Ideal (BBI) atau Indeks Massa Tubuh (IMT) untuk mengetahui berat badan pasien untuk dikategorikan karena dalam perhitungan kalori harian memiliki perhitungan berbeda-beda tergantung pada kondisi berat badan pasien. Setelah didapatkan berat badan ideal pasien, selanjutnya adalah perhitungan kebutuhan kalori basal berdasarkan jenis kelamin, usia, pekerjaan, dan berat badan. Tahap terakhir adalah perhitungan total kalori harian dan juga kebutuhan kalori karbohidrat, lemak, dan protein bagi pasien.

Proses yang kedua adalah melakukan inisialisasi parameter awal untuk perhitungan algoritme *Improved*-PSO berupa jumlah populasi, jumlah hari, nilai C_1&C_2, nilai r_1&r_2, dan batas iterasi. Lalu ditambahkan dengan perhitungan 2 parameter tambahan berupa *inertia weight* (*w*) dan *construction factor* (*K*). Parameter tersebut selanjutnya digunakan dalam melakukan tahap awal inisialisasi partikel yang dilanjutkan dengan perhitungan nilai *fitness* dengan menggunakan nilai penalti gizi dan variasi.

Proses terakhir dimulai saat masuk ke iterasi selanjutnya dengan melakukan perubahan kecepatan dan posisi hingga mendapatkan partikel akhir yang optimal dan kemudian menjadi hasil akhir berupa komposisi bahan makanan yang mencukupi kebutuhan gizi pasien Diabetes Melitus.

4.2 Deskripsi Data

Data yang digunakan dalam penelitian ini merupakan Daftar Bahan Makanan Penukar (DBMP) yang khusus ditujukan bagi penderita Diabetes Melitus. Data bahan makanan penukar tersebut berisi bahan makanan yang terbagi dalam 8 golongan dan data yang dipakai dalam penelitian ini sebanyak 7 golongan makanan. Golongan makanan tersebut antara lain sumber karbohidrat, protein

hewani, protein nabati, buah dan gula, minyak dan lemak, sayuran dan susu. Dalam daftar bahan makanan penukar tersebut memuat nama makanan, berat, Ukuran Rumah Tangga (URT), dan kandungan gizi yang terdiri dari kalori, protein, dan karbohidrat. Daftar bahan makanan penukar yang lengkap dapat dilihat pada Lampiran C dan berikut dalam Tabel 4.1 merupakan contoh golongan makanan yang termasuk dalam golongan II yaitu protein hewani lengkap dengan kandungan gizi, berat, dan ukuran rumah tangganya.

Tabel 4.1 Daftar Bahan Makanan Penukar Golongan II (Protein Hewani)

Bahan Makanan	Berat (gr)	URT	Kalori	Protein	Lemak
Ayam Tanpa Kulit	40	1 ptg sdg	50 kcal	7 g	2 g
Daging Kerbau	35	1 ptg sdg	50 kcal	7 g	2 g
Ikan Segar	50	1 ptg sdg	50 kcal	7 g	2 g
Ikan Kering	15	1 ptg kcl	50 kcal	7 g	2 g
Bakso	100	10 bj kcl	75 kcal	7 g	5 g
Daging Kambing	40	1 ptg sdg	75 kcal	7 g	5 g
Daging Sapi	35	1 ptg sdg	75 kcal	7 g	5 g
Hati Ayam	30	1 bh sdg	75 kcal	7 g	5 g
Otak	65	1 ptg bsr	75 kcal	7 g	5 g
Telur Ayam	50	1 btr	75 kcal	7 g	5 g
Udang Segar	35	1 ekor sdg	75 kcal	7 g	5 g
Telur Puyuh	60	5 btr kcl	75 kcal	7 g	5 g
Babat	40	2 ptg sdg	75 kcal	7 g	5 g
Bebek	45	1 ptg sdg	150 kcal	7 g	13 g
Cornet Beef	45	3 sdm	150 kcal	7 g	13 g
Ayam dengan Kulit	55	1 ptg sdg	150 kcal	7 g	13 g
Sosis	50	$^1/_2$ ptg sdg	150 kcal	7 g	13 g
Kuning Telur	45	2 btr	150 kcal	7 g	13 g

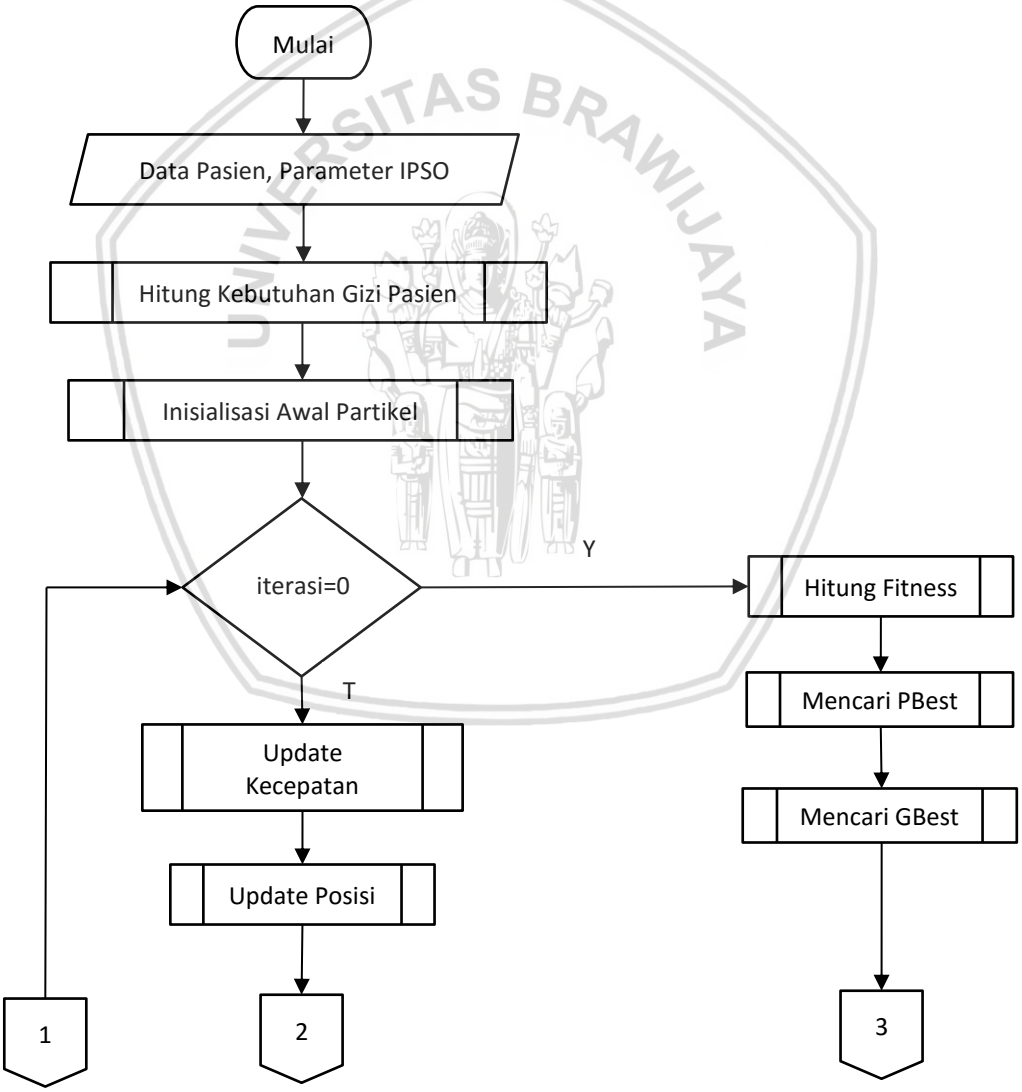
(Sumber: Daftar Bahan Makanan Penukar (DBMP), DietIndo)

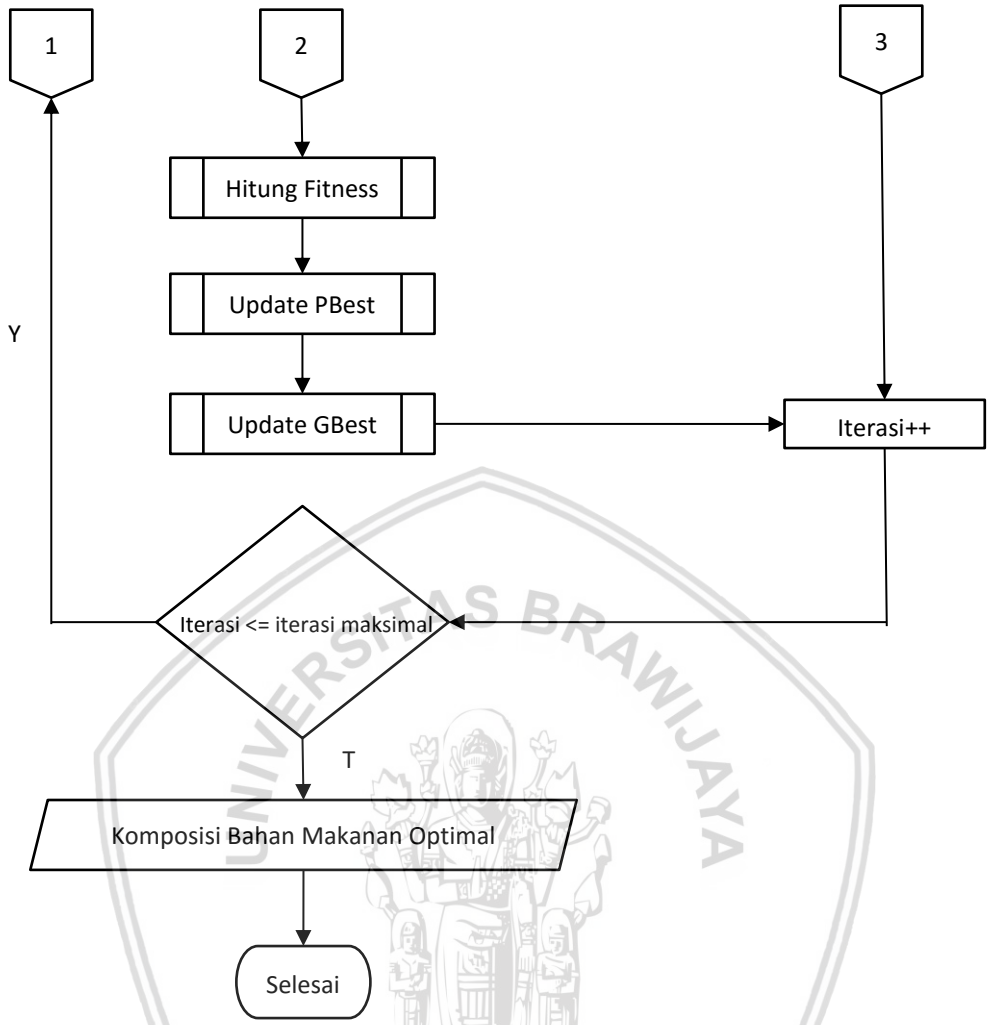
Bagi pasien Diabetes Melitus tanpa komplikasi, anjuran kebutuhan gizi dapat dipenuhi dengan mengkonsumsi makanan yang berupa sumber karbohidrat, protein hewani, protein nabati, sayuran, dan lemak dalam satu kali makan. Dalam satu hari, dianjurkan untuk makan sebanyak 3 kali dengan diselingi oleh 3 kali cemilan dengan aturan setiap kali makan, selang 3 jam dapat mengkonsumsi camilan dan 2 jam kemudian kembali makan.

Data yang terakhir berupa data pasien Diabetes Melitus tanpa komplikasi yang berusia di atas 40 tahun yang digunakan sebagai data untuk pengujian sistem. Data pasien didapatkan dari Ahli Gizi dan dapat dilihat pada Lampiran B.

4.3 Tahap Algoritme *Improved Particle Swarm Optimization*

Pada sub bab ini menjelaskan tentang tahap-tahap dalam melakukan implementasi algoritme *Improved*-PSO dalam optimasi kebutuhan makanan bagi penderita penyakit Diabetes Melitus. Tahapan algoritme *Improved*-PSO ini dilakukan secara terus menerus hingga mencapai iterasi maksimal atau kondisi berhenti telah dipenuhi. Setelah mencapai iterasi maksimal, maka proses telah selesai dan hasil yang keluar dari sistem berupa komposisi makanan yang optimal yang didapatkan dari kandidat solusi. Berikut langkah-langkah dalam algoritme *Improved*-PSO dijelaskan pada Gambar 4.1.





Gambar 4.1 Diagram Alir Tahap Algoritme *Improved*-PSO

Berdasarkan diagram alir di atas, dapat dilihat bahwa terdapat beberapa parameter yang dibutuhkan dalam perhitungan algoritme *Improved*-PSO. Setelah memasukkan parameter yang dibutuhkan, terlebih dahulu menghitung kebutuhan gizi pasien yang digunakan sebagai acuan dalam menentukan bahan makanan yang optimal. Dilanjutkan dengan proses inisialisasi populasi awal, menghitung *fitness*, menentukan *PBest* awal, dan *GBest* awal untuk iterasi ke-0. Dalam iterasi berikutnya dilakukan proses *update* kecepatan dan posisi sebelum menghitung *fitness*nya kembali. Setelah *fitness* baru didapatkan, maka dilanjutkan dengan melakukan *update PBest* dan *GBest* yang baru untuk iterasi tersebut. Proses ini berlanjut hingga mencapai batas iterasi maksimal. Akhirnya, didapatkan hasil bahan makanan yang paling optimal dari perhitungan algoritme *Improved*-PSO ini.

4.3.1 *Input* Data Pasien

Pada proses awal perhitungan, pengguna melakukan *input* data berupa data pribadi pasien yang berupa nama, berat badan, tinggi badan, jenis kelamin, dan pekerjaan. Data pasien tersebut digunakan dalam menentukan berat badan ideal

(BBI) dan menentukan kebutuhan kalori harian. Pengguna juga menentukan parameter *Improved*-PSO berupa jumlah populasi, iterasi maksimum. Untuk parameter tambahan seperti nilai C_1&C_2, *inertia weight*, dan *constriction factor* dihitung kemudian oleh sistem sebagai parameter tambahan.

Berikut adalah contoh data pribadi pasien penderita Diabetes Melitus dan parameter *Improved*-PSO yang akan digunakan sebagai perhitungan, dapat dilihat pada Tabel 4.2 dan Tabel 4.3.

Tabel 4.2 Data Pasien

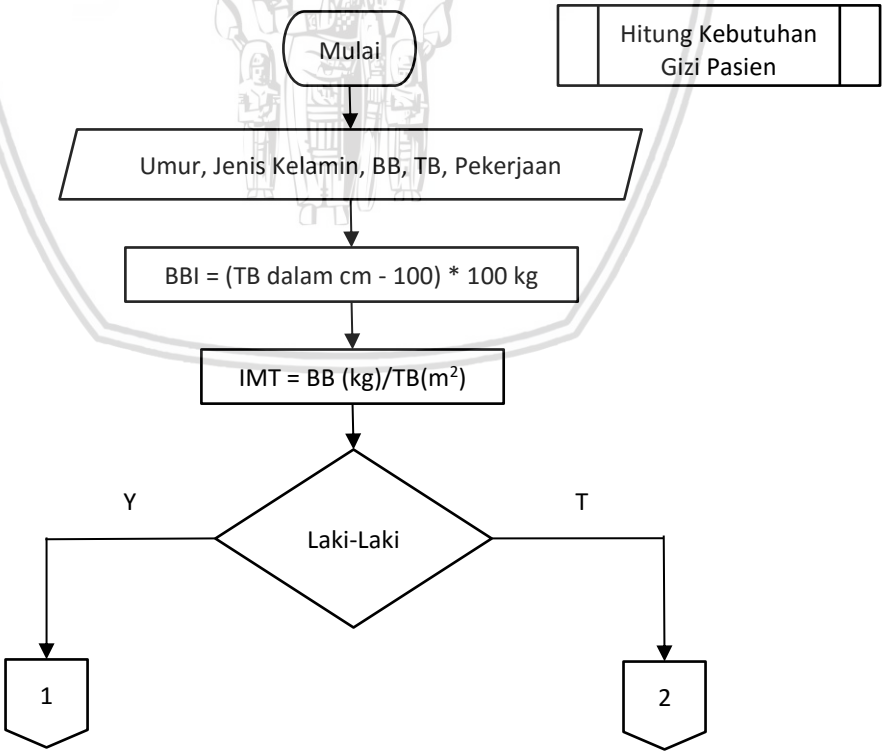
Nama	Usia	Jenis Kelamin	BB (kg)	TB (cm)	Pekerjaan
Pasien 1	50 tahun	Laki-Laki	80 kg	170 cm	Guru

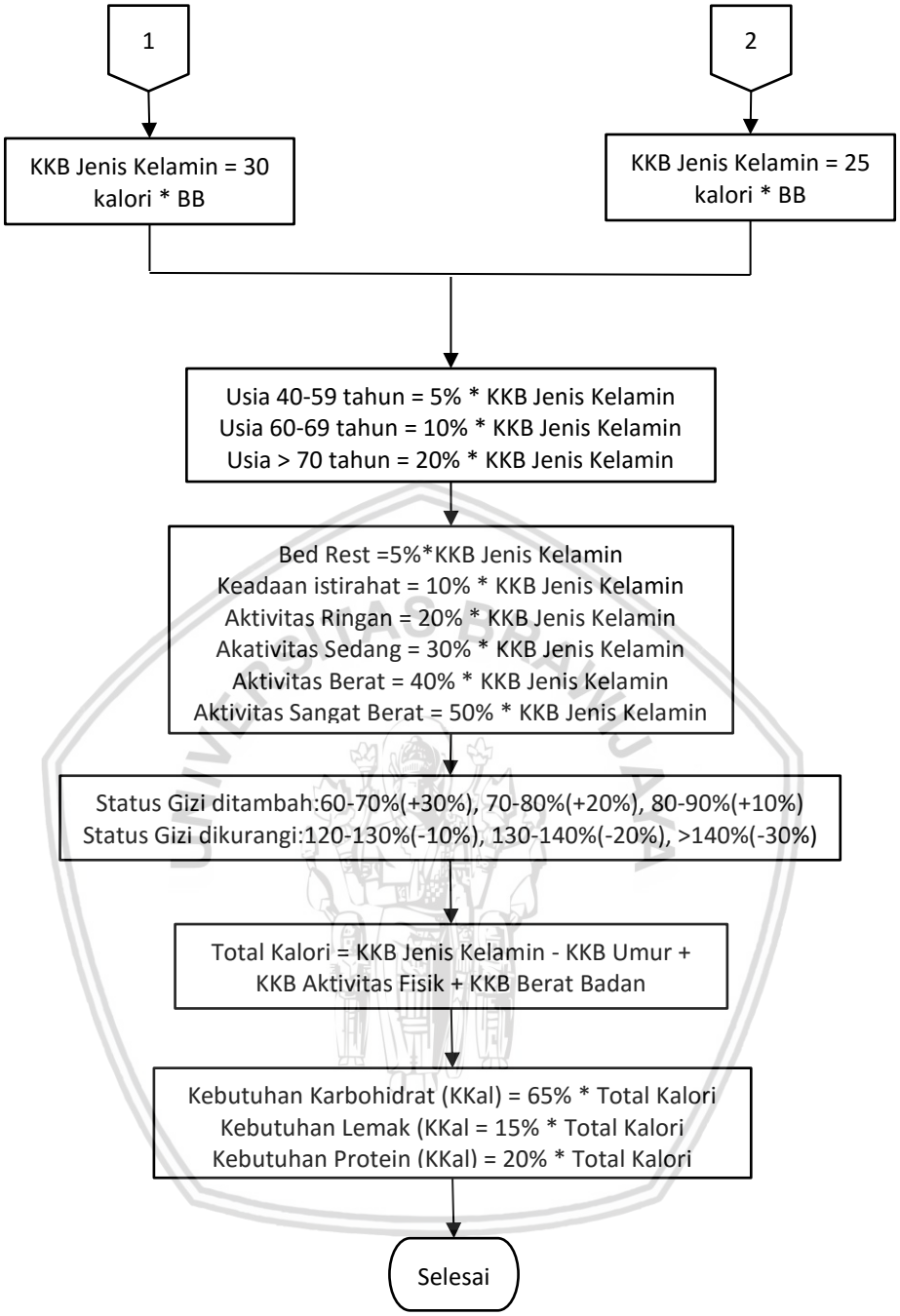
Tabel 4.3 Parameter *Improved*-PSO

Jumlah Populasi	Iterasi Maksimal
2	2

4.3.2 Perhitungan Kebutuhan Kalori Penderita Diabetes Melitus

Data pasien tersebut selanjutnya dihitung dan diproses dalam menentukan kebutuhan kalori harian. Kebutuhan kalori harian mencakup kebutuhan kalori, kebutuhan karbohidrat, kebutuhan protein, dan kebutuhan lemak. Dalam Gambar 4.2 merupakan diagram alir dalam perhitungan kebutuhan kalori harian pasien.





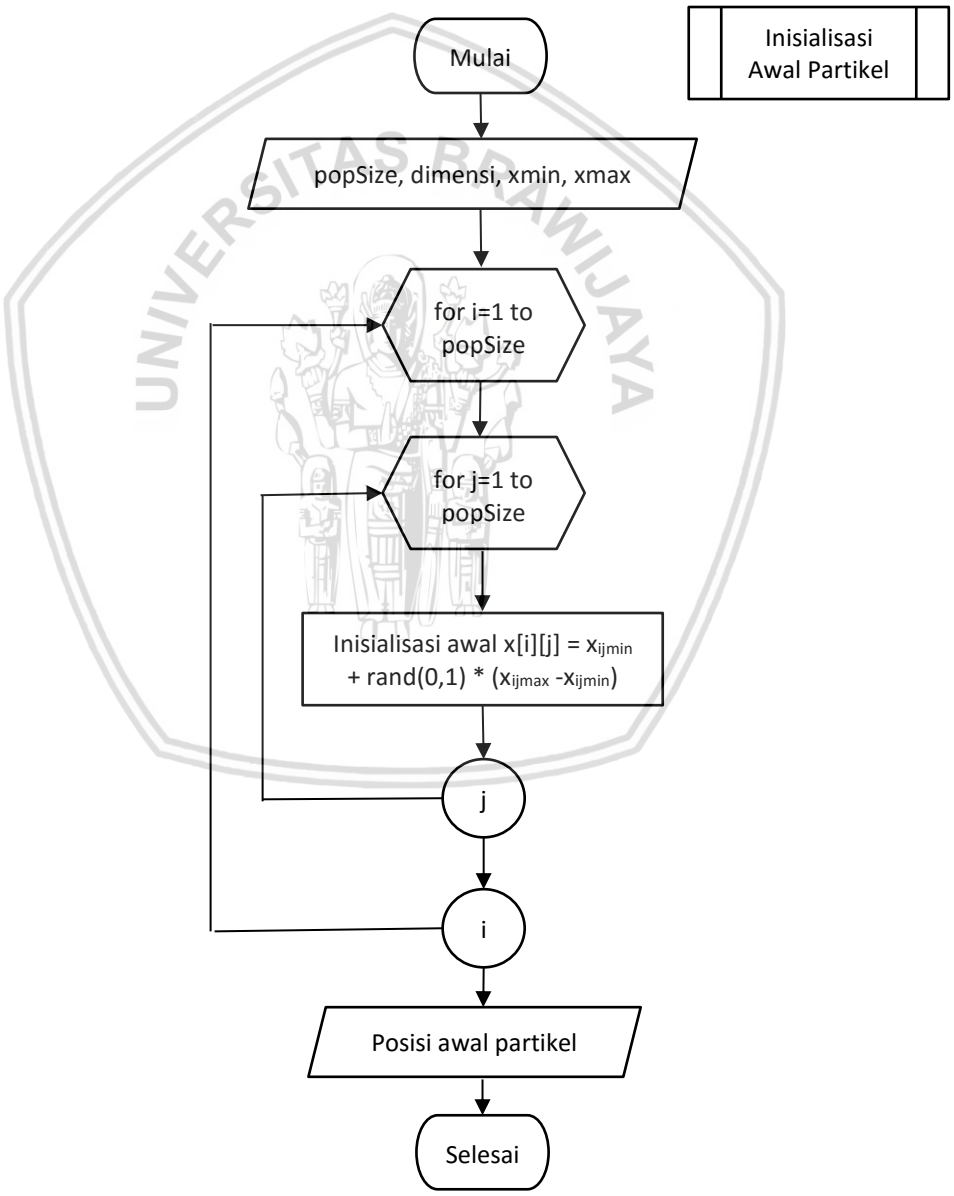
Gambar 4.2 Diagram Alir Perhitungan Kebutuhan Kalori Harian Pasien Diabetes Melitus

Dalam perhitungan kebutuhan kalori harian, dihitung berat badan ideal terlebih dahulu yang digunakan sebagai acuan dalam menghitung Kebutuhan Kalori Basal (KKB) di proses selanjutnya. Lalu perhitungan IMT untuk menentukan keadaan tubuh pasien apakah termasuk dalam keadaan kurus, normal, atau obesitas. Selanjutnya perhitungan KKB dibagi 2 berdasarkan jenis kelamin. Dilanjutkan dengan perhitungan KKB berdasarkan usia, jenis pekerjaan, kondisi tubuh dan akhirnya didapatkan total kalori harian pasien. Perhitungan kebutuhan

karbohidrat, lemak, dan protein juga diperlukan dengan memanfaatkan nilai dari kebutuhan kalori harian pasien.

4.3.3 Inisialisasi Awal Partikel

Inisialisasi awal partikel merupakan tahap awal dalam membentuk populasi yang bertindak sebagai kandidat solusi dengan jumlah serta nilainya yang ditentukan oleh nilai-nilai parameter yang ada sebelumnya. Inisialisasi posisi dan kecepatan awal dilakukan dengan metode *Real-Code* PSO yang dibangkitkan sesuai dengan dimensi dan *popSize* yang telah ditentukan. Nilai posisi partikel di sini merepresentasikan nomor bahan makanan yang tertera pada tabel masing-masing bahan makanan sesuai dengan golongannya. Dalam Gambar 4.3 merupakan diagram alir dalam proses inisialisasi partikel awal.

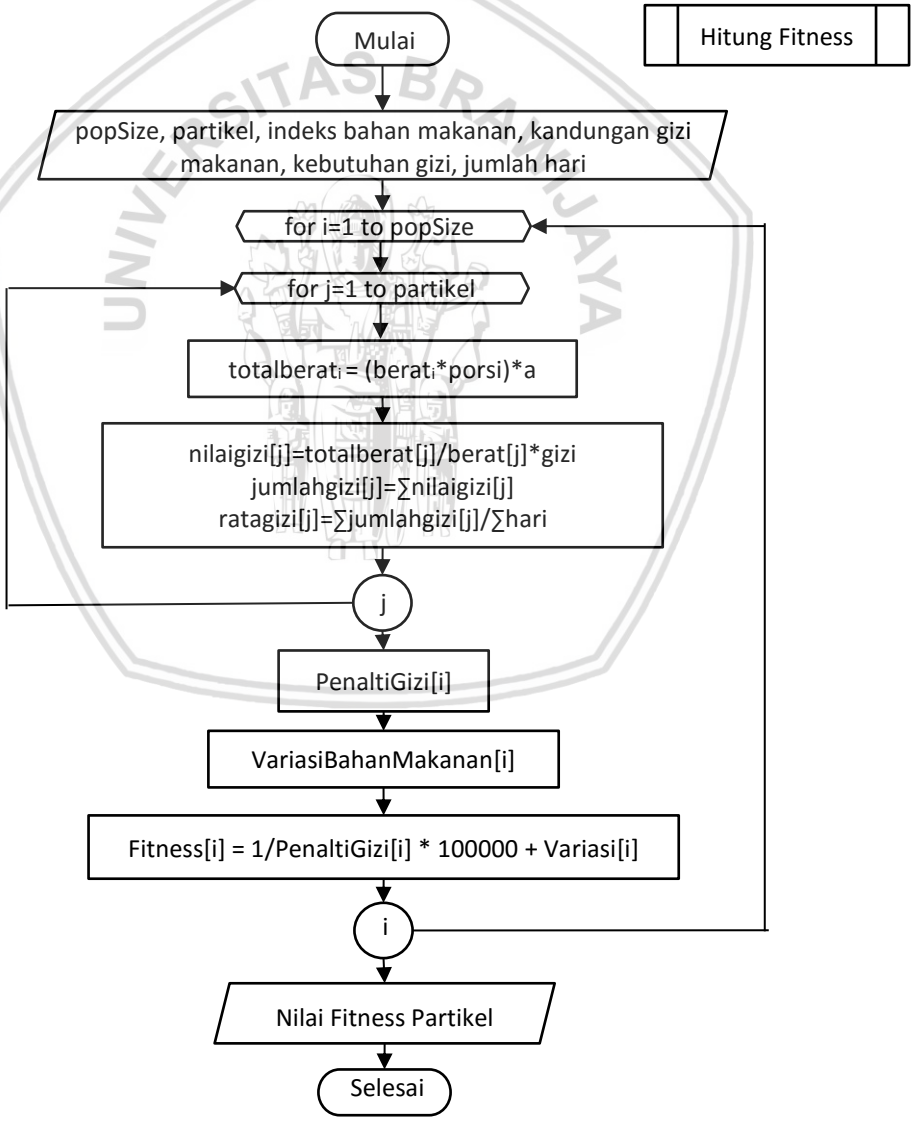


Gambar 4.3 Diagram Alir Inisialisasi Awal Partikel

Inisialisasi awal di tiap partikelnya memiliki nilai batas atas dan bawah yang berbeda tergantung pada banyaknya bahan makanan di tiap golongannya. Jumlah dimensi dalam satu populasi adalah sebanyak 36 dimensi dengan rincian terdapat 5 golongan makanan yang dikonsumsi untuk 3 kali dalam sehari dengan selang 3 kali camilan dengan 2 golongan makanan dan perhitungan sebanyak 2 hari.

4.3.4 Perhitungan *Fitness*

Setelah mendapatkan nilai dimensi untuk masing-masing partikel dan dikonversikan kedalam daftar bahan makanan, maka dilakukan perhitungan nilai *fitness*. Nilai *fitness* menggambarkan sebuah kandidat dari solusi yang ditawarkan. Nilai *fitness* yang tinggi membuatnya dapat dipilih sebagai solusi terbaik yang ditawarkan. Dalam perhitungan nilai *fitness* ini diperlukan nilai dimensi yang telah dikonversi menjadi bahan makanan sebagai nilai partikel. Berikut dalam Gambar 4.4 merupakan diagram alir perhitungan nilai *fitness*.

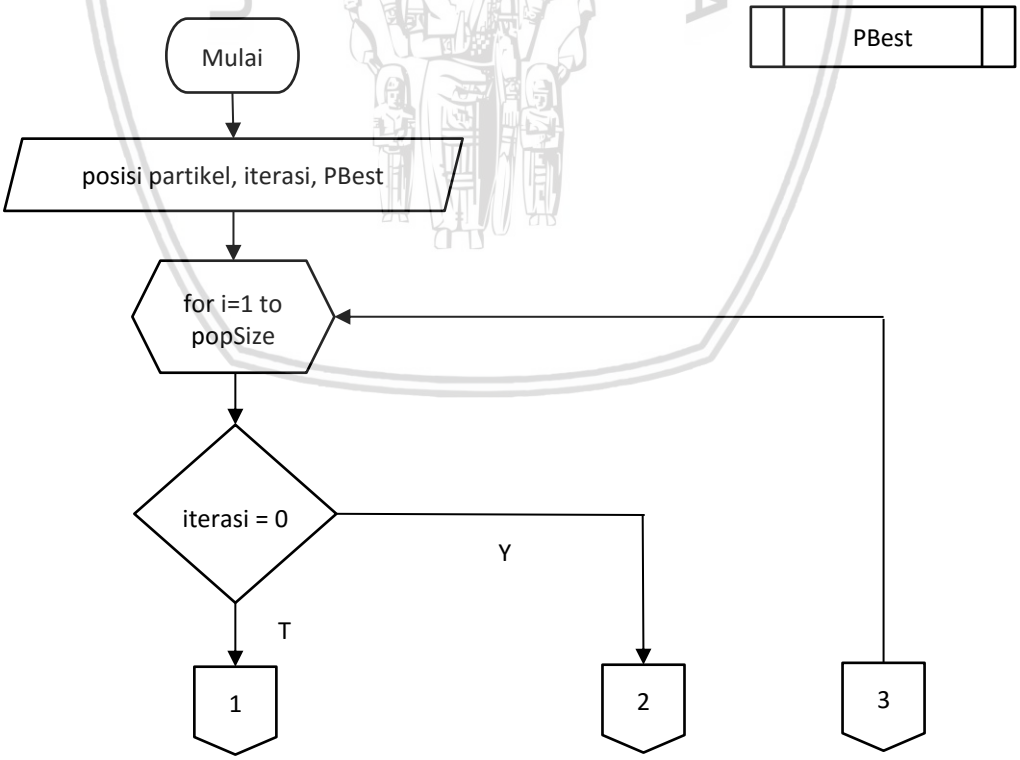


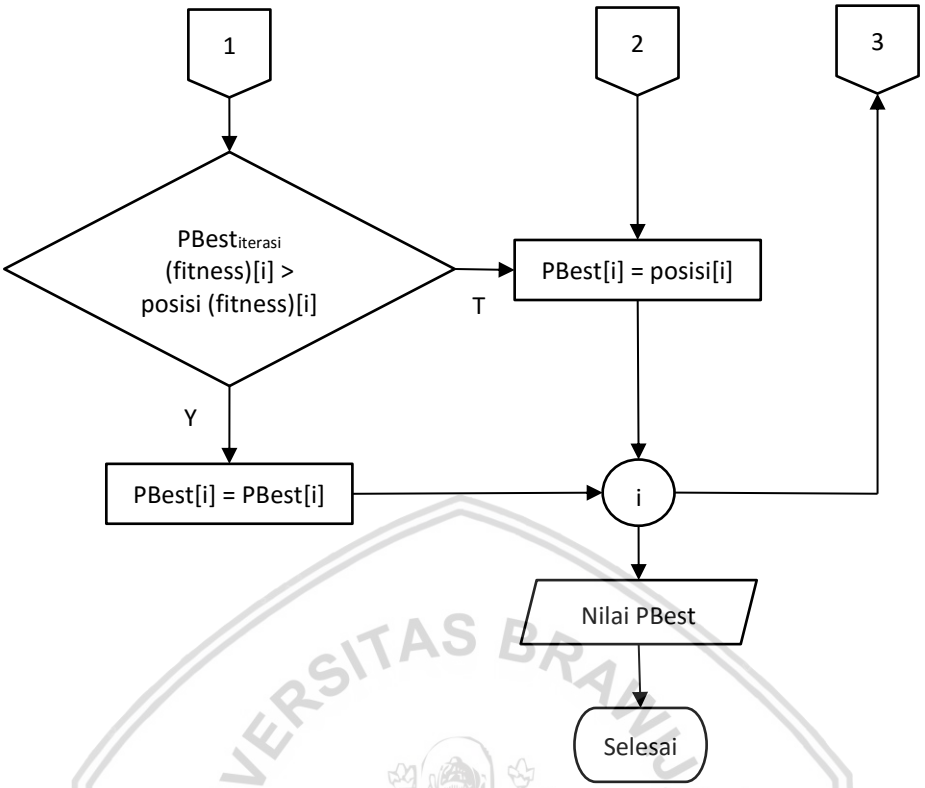
Gambar 4.4 Diagram Alir Perhitungan *Fitness*

Setelah mendapatkan nilai dimensi bahan makanan di tiap partikelnya, dilanjutkan dengan menghitung jumlah berat bahan makanan yang tujuannya adalah mengetahui banyaknya bahan makanan yang dapat dikonsumsi oleh pasien berdasarkan porsi makan dan jadwal makan pasien. Rata-rata kandungan gizi bahan makanan juga harus dihitung dengan cara menghitung besarnya kandungan kalori, karbohidrat, protein, dan lemak yang ada dalam bahan makanan. Penalti gizi kemudian dihitung sesuai dengan Persamaan 2.38 serta menghitung jumlah variasi makanan tiap populasi. Nilai penalti gizi dan jumlah variasi itulah yang digunakan dalam perhitungan nilai *fitness*.

4.3.5 Mencari *PBest*

Setelah mendapatkan seluruh nilai *fitness* dari partikel, langkah selanjutnya adalah menentukan nilai lokal terbaik dalam partikel yang selanjutnya disebut *PBest* yang merepresentasikan posisi terbaik dari suatu partikel. Dalam menentukan *PBest* ini dibagi dalam dua kondisi berbeda. Pertama, penentuan *PBest* pada iterasi awal, nilainya disamakan dengan posisi awal dikarenakan belum adanya perubahan kecepatan ataupun perubahan nilai *fitness*. Kondisi kedua adalah saat mulai masuk ke iterasi pertama hingga seterusnya, maka mulai masuk pada proses *update PBest*. Proses ini dilakukan dengan cara membandingkan nilai *fitness* suatu iterasi dengan iterasi yang sebelumnya, lalu dari kedua *fitness* tersebut dipilih partikel dengan *fitness* terbesar sebagai *PBest* yang baru. Berikut diagram alir dalam penentuan *PBest* yang terdapat pada Gambar 4.5.

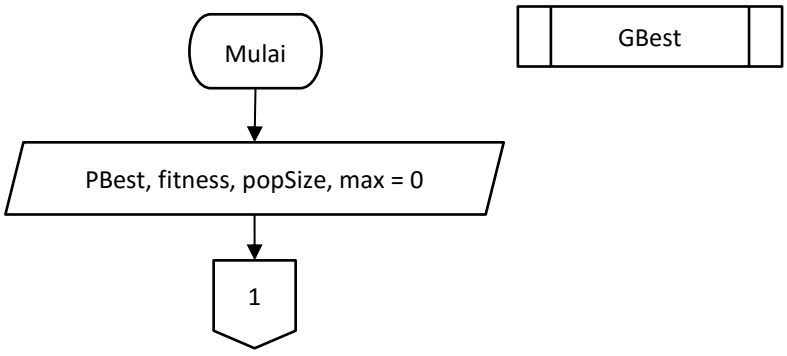


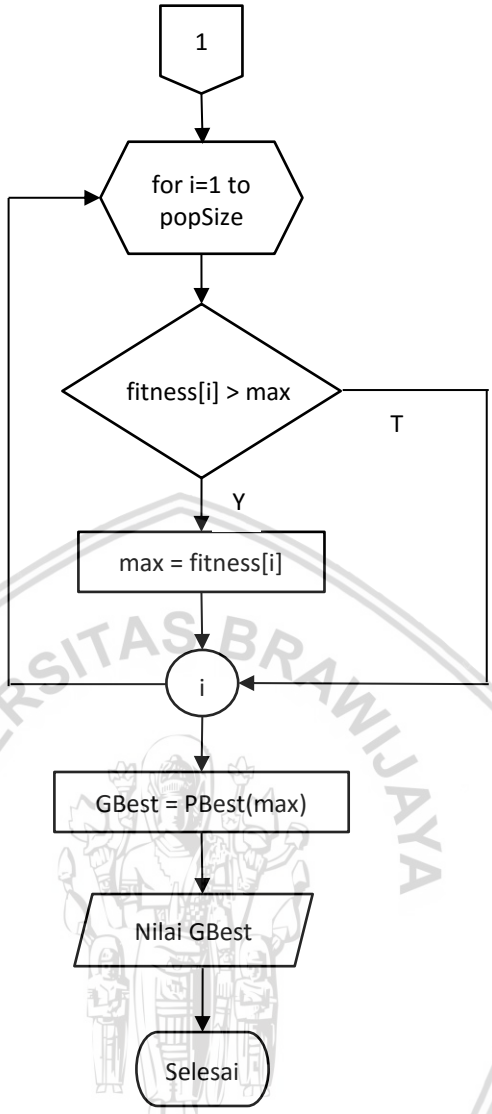


Gambar 4.5 Diagram Alir Pencarian Posisi Lokal Terbaik (*PBest*)

4.3.6 Mencari *GBest*

Langkah selanjutnya setelah mendapatkan posisi lokal (*PBest*) terbaik, maka dilanjutkan dengan mencari posisi global terbaik atau disebut *GBest*. *GBest* merupakan representasi nilai *fitness* terbaik yang didapatkan dari seluruh partikel yang ada pada populasi dan merupakan hasil dari solusi paling optimal dari seluruh kandidat solusi yang ditawarkan selama satu iterasi. Di tiap iterasinya disimpan nilai fitness terbesar sebagai *GBest* sementara dan perhitungan terus dilakukan hingga memasuki iterasi terakhir. Nilai *GBest* sementara terus diperbaiki apabila terdapat nilai *fitness* yang lebih besar lagi di tiap iterasinya. Nilai *GBest* pada akhir iterasi merepresentasikan hasil akhir perhitungan algoritme dan dikonversikan ulang kedalam tabel bahan makanan sebagai hasil optimasi. Berikut diagram alir dalam penentuan *GBest* yang terdapat pada Gambar 4.6.



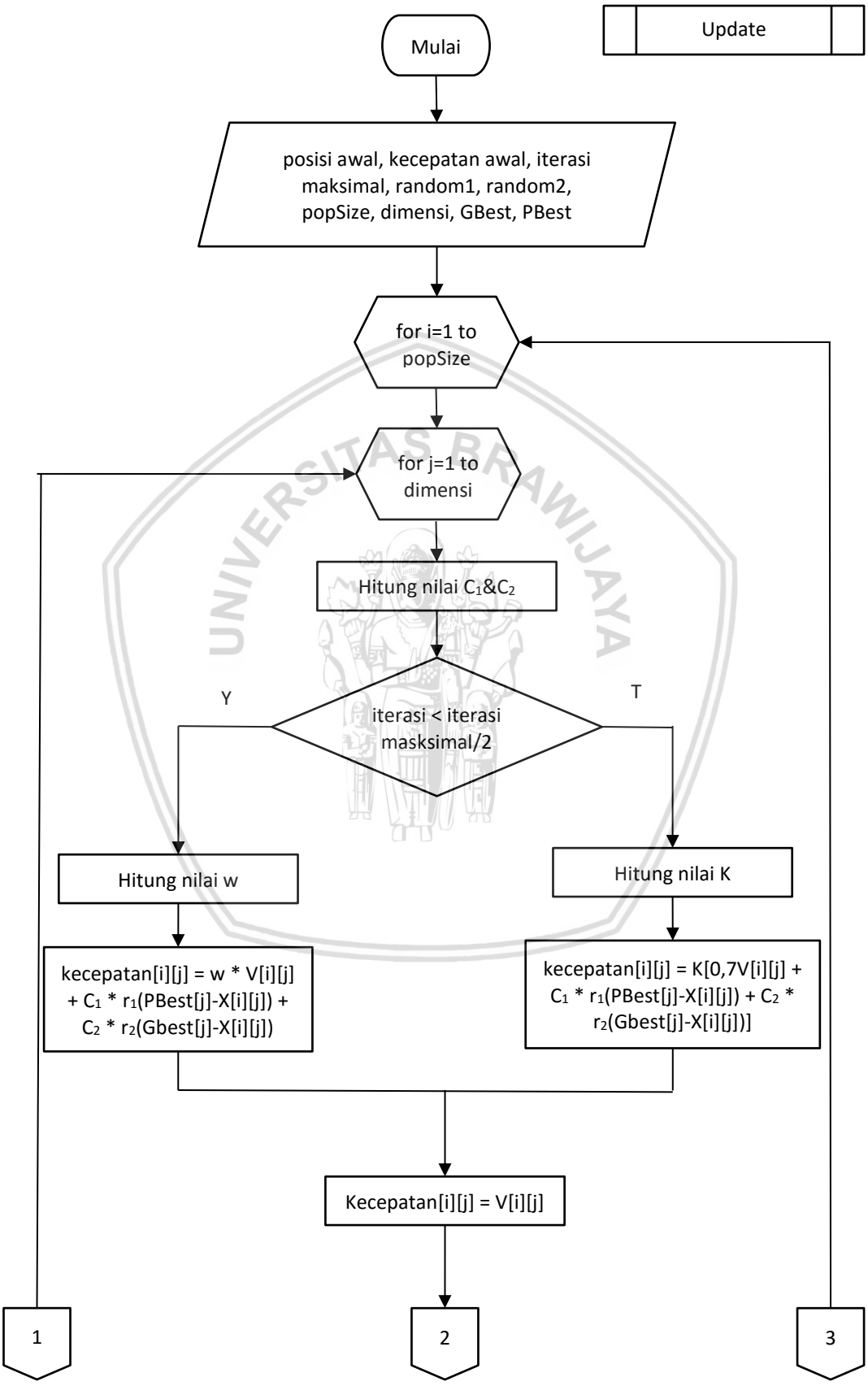


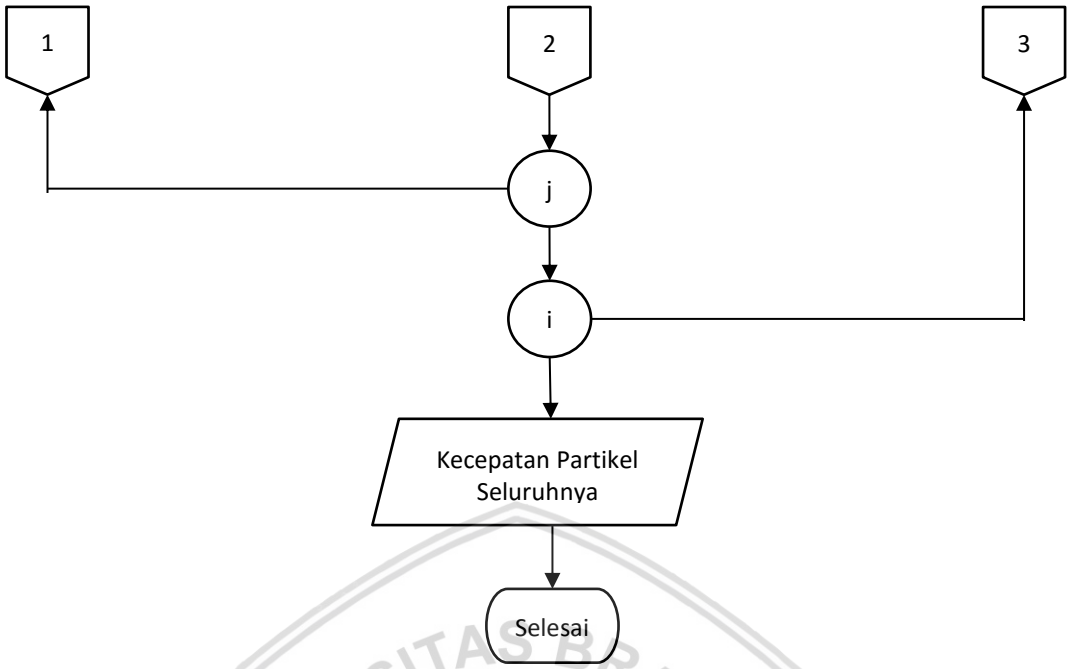
Gambar 4.6 Diagram Alir Pencarian Posisi Global Terbaik (*GBest*)

4.3.7 *Update* Kecepatan

Setelah membangun populasi dan partikel, maka dilanjutkan dengan melakukan *update* kecepatan partikel. Tahap ini mulai dilakukan apabila proses perhitungan telah masuk pada iterasi pertama hingga mencapai batas iterasi. *Update* kecepatan ini dilakukan untuk menentukan perpindahan arah partikel berdasarkan pada Persamaan 2.32. Dalam prosesnya, *update* kecepatan dibagi menjadi dua kondisi sesuai dengan algoritma *Improved*-PSO yang diterapkan. Kondisi pertama adalah saat iterasi kurang dari setengah iterasi maksimal, maka parameter yang digunakan dalam perhitungan *update* kecepatan adalah *inertia weight* (*w*). Kondisi kedua adalah apabila iterasi telah memasuki jumlah lebih dari setengah iterasi maksimal maka parameter yang digunakan dalam perhitungan *update* kecepatan adalah *constriction factor* (*K*). Untuk mencari nilai *w* dan *K* sesuai dengan Persamaan 2.29 & 2.30, lalu dalam perhitungan kecepatan ini menerapkan juga TVAC dalam pencarian nilai C_1&C_2 seperti pada Persamaan 2.31.

Berikut diagram alir dalam proses *update* kecepatan yang terdapat pada Gambar 4.7.

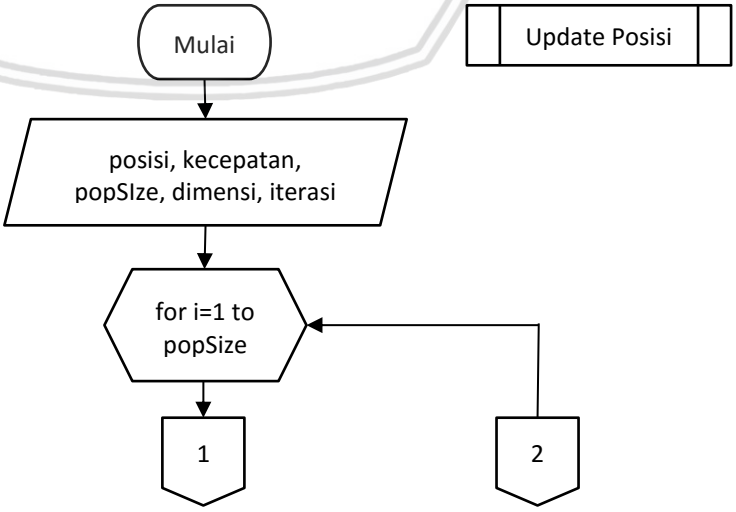


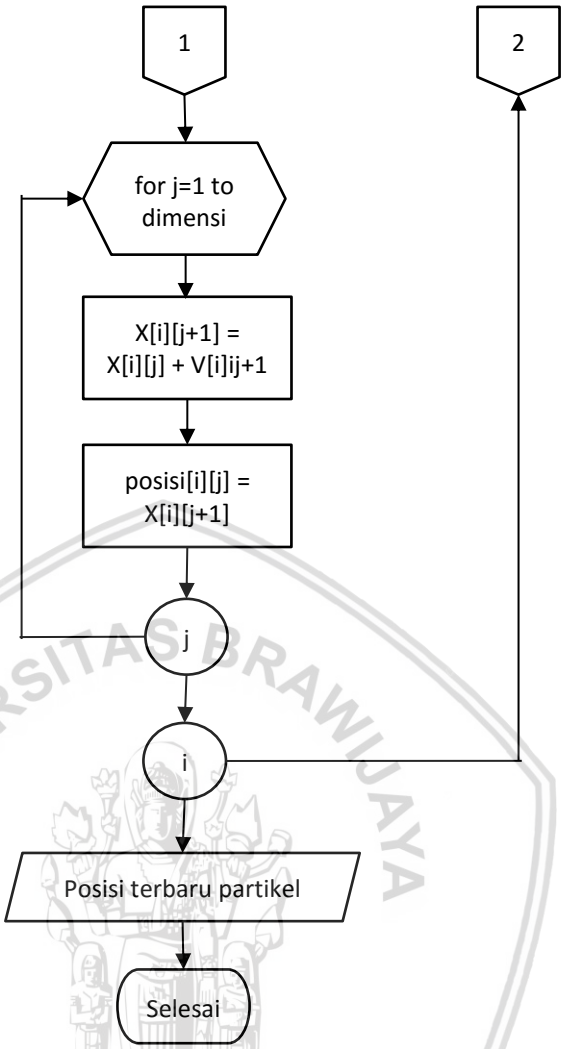


Gambar 4.7 Diagram Alir Proses *Update* Kecepatan

4.3.8 *Update* Posisi

Proses *update* posisi dilakukan tepat setelah proses *update* kecepatan sesuai dengan Persamaan 2.44 dengan cara menjumlah nilai posisi partikel di iterasi sebelumnya dengan nilai kecepatan yang baru. Nantinya didapatkan posisi partikel terbaru sebagai bobot partikel yang digunakan pada iterasi selanjutnya. Dalam *update* posisi ini juga dilihat apabila posisi terbaru partikel melebihi nilai batas atas partikel, maka nilainya dirubah menjadi batas atas partikel. Sama halnya apabila posisi yang terbaru memiliki nilai yang kurang dari nilai batas bawah partikel, maka nilainya dirubah menjadi batas bawah partikel. Berikut diagram alir dalam proses *update* posisi yang terdapat pada Gambar 4.8.





Gambar 4.8 Diagram Alir Proses *Update* Posisi

4.4 Perhitungan Manual untuk Penyelesaian Masalah dengan Menggunakan Algoritme *Improved Particle Swarm Optimization*

Sub bab ini berisi mengenai perhtiungan manual dalam penyelesaian optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita Diabetes Melitus dengan menggunakan algoritme *Improved*-PSO. Berikut adalah parameter yang digunakan dalam pengujian ini:

- Jumlah populasi : 2
- Nilai r_1 : 0,9342 (dibuat konstan hanya untuk perhitungan manual)
- Nilai r_2 : 0,3963 (dibuat konstan hanya untuk perhitungan manual)
- Iterasi maksimal : 2

4.4.1 Hitung Kebutuhan Kalori Penderita Diabetes Melitus

Berdasarkan pada data pasien di Tabel 4.2 dan dengan menggunakan Persamaan 2.4 sampai 2.28, berikut contoh perhitungan kebutuhan gizi pasien:

- BBI = 90%*(170 - 100)*1 kg = 63kg
- IMT = 87/1,7 = 27,68 (Obes I)
- KKB Jenis Kelamin (Laki-laki) = 30*BB = 2400
- KKB Umur (40-59th) = 5%*KKB Jenis Kelamin = 120
- KKB Pekerjaan Ringan (Guru) = 20%*KKB Jenis Kelamin = 480
- Status Gizi = 80/63*100 = 126,98413
- KKB Berat Badan = -10%*KKB Jenis Kelamin = -240
- Total Kebutuhan Kalori = 2520 kalori
- Kebutuhan karbohidrat = 65%*total kalori/4 = 409,5 gr
- Kebutuhan protein = 15%*total kalori/4 = 94,5 gr
- Kebutuhan lemak = 20%*total kalori/9 = 56 gr

4.4.2 Inisialisasi Awal Partikel

Tahap berikutnya merupakan inisialisasi awal dengan membangkitkan posisi dan kecepatan partikel. Dalam perhitungan ini digunakan 2 partikel yang menunjukkan perhitungan untuk 2 hari dan masing-masing partikel memiliki dimensi sebanyak 36.

Dalam 36 dimensi tersebut, terdapat 5 golongan makanan utama yang dapat dikonsumsi dalam 1 hari dengan 3 kali makan dengan diselingi oleh 3 kali camilan dengan menggunakan 2 golongan makanan sampingan. Berikut contoh pembentukan dimensi seperti pada Tabel 4.4.

Tabel 4.4 Tabel Pembentukan Dimensi

Hari Ke-1																	
Pagi					CM	Siang					CM	Malam					CM
SK	PH	PN	S	L	C	SK	PH	PN	S	L	C	SK	PH	PN	S	L	C
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Hari Ke-2																	
Pagi					CM	Siang					CM	Malam					CM
SK	PH	PN	S	L	C	SK	PH	PN	S	L	C	SK	PH	PN	S	L	CM
1	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36

Keterangan:

- SK : Sumber Karbohidrat
- PH : Protein Hewani
- PN : Protein Nabati
- S : Sayuran
- L : Lemak

- C : Camilan (Susu, Buah, dan Gula)

Masing-masing golongan makanan memiliki jenis yang bermacam-macam yang nantinya jumlah dari jenis bahan makanan inilah yang digunakan sebagai x_{min} dan x_{max}. Dalam inisialisasi awal partikel maupun *update* posisi nantinya, x_{min} dan x_{max} ini digunakan sebagai patokan dalam pembentukan nilai dimensi. Apabila nilai dimensinya menghasilkan nilai di bawah 0, maka nilainya digantikan oleh 0 (x_{min}=0). Sebaliknya, untuk x_{max} atau batas atas memiliki nilai yang berbeda-beda pada tiap golongan. Apabila nilai dimensinya melebihi batas atas, maka nilainya disamakan dengan batas atasnya. Berikut jumlah jenis dari golongan makanan yang ditampilkan pada Tabel 4.5.

Tabel 4.5 Tabel Nilai Batas Atas Partikel

Golongan Bahan Makanan	Jumlah Bahan Makanan (Batas Atas)
Sumber Karbohidrat	21
Protein Hewani	9
Protein Nabati	9
Sayuran	47
Lemak dan Minyak	12
Camillan (Buah dan Susu)	32

Posisi awal partikel ditentukan dengan menggunakan Persamaan 2.33 dengan menggunakan nilai *random* (acak) dengan interval 0-1 dan juga nilai bobot maksimal bahan pakan yang berbeda-beda sesuai dengan golongan bahan makanan di tiap partikel. Berikut merupakan hasil perhitungan inisialisasi posisi awal partikel yang ditampilkan dalam Tabel 4.6.

Tabel 4.6 Tabel Inisialisasi Posisi Awal Partikel

Partikel	Nilai Dimensi					
	SK	PH	PN	S	L	C
X1	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166
	17.02194	8.740330515	3.64618	25.94335	8.223065	31.20912
	1.073075	3.638113825	8.097785	8.567124	7.313755	10.46032
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745
X2	6.400614	4.12644985	5.582587	11.59122	8.921589	25.42772
	6.868579	3.8306408	8.355252	12.05599	9.906731	27.45027
	17.8424	2.463164847	4.726547	14.7141	9.303328	8.548397
	20.55493	7.80920979	1.595176	23.65348	1.147892	31.46644
	9.494672	2.0490959	1.542118	24.00386	1.896984	17.01349
	6.353222	5.00930191	4.485197	26.1172	10.70894	27.45567

Setelah melakukan inisialisasi awal posisi seperti pada tabel di atas, berikutnya adalah inisialisasi kecepatan awal di mana seluruh nilainya adalah 0. Berikut nilai kecepatan awal partikel ditampilkan pada Tabel 4.7.

Tabel 4.7 Tabel Kecepatan Awal Partikel

Partikel	Kecepatan Awal					
	SK	PH	PN	S	L	C
V1	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
V2	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0

4.4.3 Perhitungan *Fitness*

Langkah selanjutnya adalah melakukan perhitungan nilai *fitness* di tiap partikelnya, namun sebelumnya diharuskan untuk melakukan proses perhitungan lain. Berikut merupakan tahap dalam perhitungan *fitness*:

1. Konversi Indeks Bahan Makanan

Setelah mendapatkan nilai di tiap partikelnya, selanjutnya dilakukan konversi indeks bahan makanan yang sesuai dengan hasil pembangkitan awal posisi. Pada daftar bahan makanan terdapat nomor indeks di tiap bahannya yang dapat dilihat pada Lampiran C. Berikut merupakan contoh daftar bahan makanan golongan protein hewani beserta indeksnya pada Tabel 4.8 dan pada Tabel 4.9 menunjukkan hasil konversi kedalam indeks bahan makanan pada partikel *X1*.

Tabel 4.8 Contoh Daftar Bahan Makan Beserta Indeks

Indeks ke-	Bahan Makanan	Berat (gr)	URT	Kalori	Protein	Lemak
1	Ayam Tanpa Kulit	40	1 ptg sdg	50 kcal	7 g	2 g
2	Daging Kerbau	35	1 ptg sdg	50 kcal	7 g	2 g
3	Ikan Segar	50	1 ptg sdg	50 kcal	7 g	2 g
4	Ikan Kering	15	1 ptg kcl	50 kcal	7 g	2 g
5	Bakso	100	10 bj kcl	75 kcal	7 g	5 g

(Sumber: Daftar Bahan Makanan Penukar (DBMP), DietIndo)

Tabel 4.9 Tabel Hasil Konversi Bahan Makanan

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Jumlah Bahan Makanan	Nilai Partikel	Nama Bahan Makanan
1	Pagi	SK	21	9	Mie Basah
		PH	9	9	Udang Segar
		PN	9	2	Tahu
		S	47	46	Selada Air
		L	12	4	Minyak Jagung
	Camilan	C	32	5	Nanas
	Siang	SK	21	18	Tepung Sagu
		PH	9	1	Ayam Tanpa Kulit
		PN	9	8	Kacang Merah
		S	47	4	Daun Pepaya
		L	12	3	Minyak Biji Bunga Matahari
	Camilan	C	32	32	Susu Kerbau
	Malam	SK	21	17	Tepung Hungkwe
		PH	9	9	Udang Segar
		PN	9	4	Kacang Hijau
		S	47	26	Daun Pakis
		L	12	8	Mentega
	Camilan	C	32	31	Susu Kedelai
2	Pagi	SK	21	1	Nasi
		PH	9	4	Ikan Kering
		PN	9	8	Kacang Merah
		S	47	9	Daun Talas
		L	12	7	Minyak Zaitun
	Camilan	C	32	10	Jambu Air
	Siang	SK	21	20	Tepung Terigu
		PH	9	3	Ikan Segar
		PN	9	2	Tahu

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Jumlah Bahan Makanan	Nilai Partikel	Nama Bahan Makanan
		S	47	23	Tauge Kacang Hijau
		L	12	3	Minyak Biji Bunga Matahari
	Camilan	C	32	31	Susu Kedelai
	Malam	SK	21	14	Krakers
		PH	9	5	Bakso
		PN	9	3	Oncom
		S	47	9	Daun Talas
		L	12	6	Minyak Kacang Tanah
	Camilan	C	32	16	Blewah

2. Menghitung Berat Bahan Makanan

Berat bahan makanan dihitung dengan tujuan mengetahui berat bahan yang sesuai dengan porsi ideal bagi pasien dengan menggunakan Persamaan 2.34. Berikut merupakan contoh perhitungan berat bahan makanan dengan mengambil berat asli bahan makanan pada partikel *X1*:

- $totalberat_{miebasah} = (berat_{miebasah} * takaranporsi_{SK}) * a_{pagi}$

  = (100gr * 1.5) * 25% = 37,5gr

- $totalberat_{tepungsagu} = (berat_{tepungsagu} * takaranporsi_{SK}) * a_{siang}$

  = (40gr * 7) * 25% = 70gr

- $totalberat_{bakso} = (berat_{bakso} * takaranporsi_{PH}) * a_{malam}$

  = (100gr * 10) * 25% = 250gr

Dari hasil di atas dapat dilihat bahwa berat bakso yang dibutuhkan untuk dikonsumsi adalah 250gr di saat makan malam, mie basah sebanyak 37.5gr di pagi hari, dan tepung sagu sebanyak 70gr di siang hari. Berikut dalam Tabel 4.10 menunjukkan hasil perhitungan berat bahan makanan pada *X1*.

Tabel 4.10 Kebutuhan Berat Bahan Makanan

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Nilai Partikel	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)
1	Pagi	SK	9	Mie Basah	100	37.5
		PH	9	Udang Segar	35	8.75
		PN	2	Tahu	100	25
		S	46	Selada Air	100	25

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Nilai Partikel	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)
		L	4	Minyak Jagung	5	1.25
	Camilan	C	5	Nanas	75	1.2
	Siang	SK	18	Tepung Sagu	40	70
		PH	1	Ayam Tanpa Kulit	40	10
		PN	8	Kacang Merah	15	9.375
		S	4	Daun Pepaya	100	25
		L	3	Minyak Biji Bunga Matahari	5	1.25
	Camilan	C	32	Susu Kerbau	100	5
	Malam	SK	17	Tepung Hungkwe	40	85
		PH	9	Udang Segar	35	8.75
		PN	4	Kacang Hijau	15	1.875
		S	26	Daun Pakis	100	25
		L	8	Mentega	15	3.75
	Camilan	C	31	Susu Kedelai	200	20
2	Pagi	SK	1	Nasi	100	12.5
		PH	4	Ikan Kering	15	3.75
		PN	8	Kacang Merah	15	9.375
		S	9	Daun Talas	100	25
		L	7	Minyak Zaitun	5	1.25
	Camilan	C	10	Jambu Air	110	22
	Siang	SK	20	Tepung Terigu	50	100
		PH	3	Ikan Segar	50	12.5
		PN	2	Tahu	100	25
		S	23	Tauge Kacang Hijau	100	25
		L	3	Minyak Biji Bunga Matahari	5	1.25
	Camilan	C	31	Susu Kedelai	200	20
	Malam	SK	14	Krakers	50	62.5
		PH	5	Bakso	100	250
		PN	3	Oncom	50	25
		S	9	Daun Talas	100	25
		L	6	Minyak Kacang Tanah	5	1.25
	Camilan	C	16	Blewah	100	10

3. Menghitung Rata-rata Kandungan Gizi

Dalam daftar bahan makanan penukar ini tiap bahannya memiliki kandungan gizi yang berbeda-beda, mulai dari kandungan karbohidrat, protein, dan lemak. Dalam menghitung rata-rata kandungan gizi, terlebih dahulu menghitung kandungan gizi di masing-masing bahan makanan dengan menggunakan Persamaan 2.35. Berikut merupakan contoh perhitungan kandungan gizi pada bahan makanan mie basah:

- $nilai\ gizi_{miebasah,kalori} = totalberat_{miebasah}/berat_{miebasah} * gizi_{kalori}$

  = 37.5/100 * 175 = 65,625 kalori

- $nilai\ gizi_{miebasah,protein} = totalberat_{miebasah}/berat_{miebasah} * gizi_{protein}$

  = 37.5/100 * 4 = 1,5gr

- $nilai\ gizi_{miebasah,karbohidrat} = total_{beratmiebasah}/berat_{miebasah} * gizi_{karbohidrat}$

  = 37.5/100 * 40 = 15gr

Dari perhitungan di atas dapat dilihat bahwa kandungan kalori yang didapat pasien dari bahan makanan mie basah sebesar 43,75 kalori, lalu kandungan protein sebanyak 1gr dan kandungan karbohidrat sebanyak 10gr. Setelah menghitung nilai gizi di tiap bahan makanan, langkah selanjutnya adalah menghitung jumlah total nilai gizi di tiap jenisnya dalam satu partikel dan menghitung rata-rata nilai gizinya menggunakan Persamaan 2.36 dan 2.37. Berikut merupakan contoh perhitungan total nilai gizi serta rata-rata nilai gizi dalam satu partikel:

- $jumlahGizi_{kalori} = \sum gizi_{kalori}$ = 1960,225 kalori
- $rataGizi_{kalori} = \sum jumlahGizi_{kalori}/2$ = 983,1125 kalori

Berikut dalam Tabel 4.11 adalah hasil setelah menghitung seluruh total nilai tiap kandungan gizi dan menghitung rata-rata kandungan tiap gizi pada partikel *X1*.

Tabel 4.11 Tabel Jumlah dan Rata-rata Nilai Gizi

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)	Kalori (kal)	Karbo (gr)	Protein (gr)	Lemak (gr)
1	Pagi	Mie Basah	100	37.5	65.625	15	1.5	0
		Udang Segar	35	8.75	18.75	0	1.75	1.25
		Tahu	100	25	20	2	1.5	0.75
		Selada Air	100	25	0	0	0	0
		Minyak Jagung	5	1.25	12.5	0	0	1.25
	Camilan	Nanas	75	1.275	0.85	0.204	0	0
	Siang	Tepung Sagu	40	70	306.25	70	7	0
		Ayam Tanpa Kulit	40	10	12.5	0	1.75	0.5
		Kacang Merah	15	9.375	50	5	3.75	1.875
		Daun Pepaya	100	25	12.5	2.5	0.75	0
		Minyak Biji Bunga Matahari	5	1.25	12.5	0	0	1.25
	Camilan	Susu Kerbau	100	5	7.5	0.5	0.35	0.5
	Malam	Tepung Hungkwe	40	85	371.875	85	8.5	0

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)	Kalori (kal)	Karbo (gr)	Protein (gr)	Lemak (gr)
		Udang Segar	35	8.75	18.75	0	1.75	1.25
		Kacang Hijau	15	1.875	10	1	0.75	0.375
		Daun Pakis	100	25	6.25	1.25	0.25	0
		Mentega	15	3.75	12.5	0	0	1.25
	Camilan	Susu Kedelai	200	20	12.5	1	0.7	0.6
2	Pagi	Nasi	100	12.5	21.875	5	0.5	0
		Ikan Kering	15	3.75	12.5	0	1.75	0.5
		Kacang Merah	15	9.375	50	5	3.75	1.875
		Daun Talas	100	25	12.5	2.5	0.75	0
		Minyak Zaitun	5	1.25	12.5	0	0	1.25
	Camilan	Jambu Air	110	22	10	2.4	0	0
	Siang	Tepung Terigu	50	100	350	80	8	0
		Ikan Segar	50	12.5	12.5	0	1.75	0.5
		Tahu	100	25	20	2	1.5	0.75
		Tauge Kacang Hijau	100	25	6.25	1.25	0.25	0
		Minyak Biji Bunga Matahari	5	1.25	12.5	0	0	1.25
	Camilan	Susu Kedelai	200	20	12.5	1	0.7	0.6
	Malam	Krakers	50	62.5	218.75	50	5	0
		Bakso	100	250	187.5	0	17.5	12.5
		Oncom	50	25	40	4	3	1.5
		Daun Talas	100	25	12.5	2.5	0.75	0
		Minyak Kacang Tanah	5	1.25	12.5	0	0	1.25
	Camilan	Blewah	100	10	5	1.2	0	0
Jumlah Nilai Gizi					1960.225	340.304	75.5	32.825
Rata-rata Nilai Gizi					980.1125	170.152	37.75	16.4125

4. Menghitung Penalti Gizi

Tahap selanjutnya setelah didapatkan nilai kandungan gizi tiap jenisnya, maka dilanjutkan untuk perhitungan penalti gizi. Penalti gizi dihitung dengan menggunakan rumus pada Persamaan 2.39 - 2.42 dengan cara menghitung selisih kebutuhan gizi pasien dengan nilai gizi yang dihasilkan oleh makanan. Berikut merupakan contoh perhitungan penalti gizi pada partikel *X1*:

- Penalti Kalori = |(2520 – 980,1125)| = 1539,8875 kal
- Penalti Karbohidrat = |(409,5 – 170,152)| = 239,348 gr
- Penalti Protein = |(94,5 – 37,75)| = 56,75 gr

- Penalti Lemak = |(56 – 16,4125)| = 39,5875 gr
- Total Penalti = 1875,573

Total penalti gizi yang didapatkan sebesar 1875,573 yang didapatkan dari penalti kalori, karbohidrat, protein, dan lemak. Nantinya nilai total penalti gizi ini digunakan dalam perhitungan nilai *fitness*.

5. Menghitung Total Variasi

Tahap selanjutnya adalah menghitung total variasi bahan makanan dalam satu partikel. Perhitungan variasi ini dilakukan dengan cara menjumlahkan bahan makanan yang berbeda dalam satu partikel, dengan kata lain bahan makanan yang muncul lebih dari satu kali dalam partikel diabaikan. Variasi seluruh bahan makanan di tiap partikelnya dijumlah seluruhnya. Pada Tabel 4.12 berikut berisi jumlah variasi bahan makanan pada partikel *X1*.

Tabel 4.12 Total Variasi Bahan Makanan

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Jumlah Bahan Makanan	Nilai Partikel	Nama Bahan Makanan	Variasi
1	Pagi	SK	21	9	Mie Basah	1
		PH	9	9	Udang Segar	0
		PN	9	2	Tahu	0
		S	47	46	Selada Air	1
		L	12	4	Minyak Jagung	1
	Camilan	C	32	5	Nanas	1
	Siang	SK	21	18	Tepung Sagu	1
		PH	9	1	Ayam Tanpa Kulit	1
		PN	9	8	Kacang Merah	0
		S	47	4	Daun Pepaya	1
		L	12	3	Minyak Biji Bunga Matahari	0
	Camilan	C	32	32	Susu Kerbau	1
	Malam	SK	21	17	Tepung Hungkwe	1
		PH	9	9	Udang Segar	0
		PN	9	4	Kacang Hijau	1

Hari Ke-	Waktu Makan	Golongan Bahan Makanan	Jumlah Bahan Makanan	Nilai Partikel	Nama Bahan Makanan	Variasi
		S	47	26	Daun Pakis	1
		L	12	8	Mentega	1
	Camilan	C	32	31	Susu Kedelai	0
2	Pagi	SK	21	1	Nasi	1
		PH	9	4	Ikan Kering	1
		PN	9	8	Kacang Merah	0
		S	47	9	Daun Talas	0
		L	12	7	Minyak Zaitun	1
	Camilan	C	32	10	Jambu Air	1
	Siang	SK	21	20	Tepung Terigu	1
		PH	9	3	Ikan Segar	1
		PN	9	2	Tahu	0
		S	47	23	Tauge Kacang Hijau	1
		L	12	3	Minyak Biji Bunga Matahari	0
	Camilan	C	32	31	Susu Kedelai	0
	Malam	SK	21	14	Krakers	1
		PH	9	5	Bakso	1
		PN	9	3	Oncom	1
		S	47	9	Daun Talas	0
		L	12	6	Minyak Kacang Tanah	1
	Camilan	C	32	16	Blewah	1
Total Variasi						24

6. Menghitung *Fitness*

Setelah menghitung nilai penalti gizi dan total harga, tahap selanjutnya adalah menghitung nilai *fitness* partikel. Perhitungan nilai *fitness* menggunakan Persamaan 2.42. Berikut contoh perhitungan nilai *fitness* untuk populasi awal:

- $Fitness = \frac{1}{Total\ Penalti} * 100000 + variasi$
- $Fitness(X1) = \frac{1}{1875.57} * 100000 + 24 = 77{,}317$
- $Fitness(X2) = \frac{1}{2137.59} * 100000 + 20 = 66{,}781$

Tabel 4.13 berikut ini berisi nilai *fitness* dari seluruh partikel pada iterasi ke-0:

Tabel 4.13 Nilai *Fitness* Iterasi ke-0

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
X1	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166	
	17.02194	8.740330515	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.638113825	8.097785	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745	
X2	6.400614	4.12644985	5.582587	11.59122	8.921589	25.42772	66.781
	6.868579	3.8306408	8.355252	12.05599	9.906731	27.45027	
	17.8424	2.463164847	4.726547	14.7141	9.303328	8.548397	
	20.55493	7.80920979	1.595176	23.65348	1.147892	31.46644	
	9.494672	2.0490959	1.542118	24.00386	1.896984	17.01349	
	6.353222	5.00930191	4.485197	26.1172	10.70894	27.45567	

4.4.4 Menentukan *PBest* dan *GBest* Awal

Pada iterasi ke-0, nilai *PBest* disamakan dengan posisi partikel awal dan untuk nilai *GBest* diambil dari *PBest* dengan nilai *fitness* yang tertinggi. Berikut adalah hasil pembangkitan nilai *PBest* dan *GBest* awal pada Tabel 4.14 dan 4.15.

Tabel 4.14 *PBest* Awal

PBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
PBest 1	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166	
	17.02194	8.740330515	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.638113825	8.097785	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745	
PBest 2	6.400614	4.12644985	5.582587	11.59122	8.921589	25.42772	66.781
	6.868579	3.8306408	8.355252	12.05599	9.906731	27.45027	
	17.8424	2.463164847	4.726547	14.7141	9.303328	8.548397	
	20.55493	7.80920979	1.595176	23.65348	1.147892	31.46644	

PBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
	9.494672	2.0490959	1.542118	24.00386	1.896984	17.01349	
	6.353222	5.00930191	4.485197	26.1172	10.70894	27.45567	

Tabel 4.15 *GBest* Awal

GBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
GBest	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166	
	17.02194	8.740330515	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.638113825	8.097785	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745	

4.4.5 Menghitung *Update* Kecepatan

Penentuan *GBest* awal merupakan tahap terakhir dalam inisialisasi, lalu dilanjutkan pada iterasi selanjutnya dengan diawali dengan melakukan *update* kecepatan dengan menggunakan Persamaan 2.32. Pada contoh perhitungan kali ini dilakukan iterasi maksimal sebanyak 2 kali, oleh karena itu, parameter *constriction factor* (*K*) yang digunakan dalam perhitungan *update* kecepatan dengan menggunakan Persamaan 2.26. Berikut perhitungan nilai *constriction factor* (*K*):

- $K = \frac{\cos\left(\frac{2\pi}{T_{max}} x \left(\frac{t - T_{max}}{2}\right)\right) + 2{,}428571}{4}$

  $= \frac{\cos\left(\frac{2*3.14}{2} x \left(\frac{1-2}{2}\right)\right) + 2{,}428571}{4}$

  $= 0{,}6073$

Setelah mendapatkan nilai *K*, selanjutnya adalah menghitung nilai C_1 dan C_2 dengan menggunakan Persamaan 2.31. Berikut contoh perhitungan nilai C_1 dan C_2:

- $C_1 = \left(c_{1f} - c_{1i}\right)\frac{t}{t_{max}} + c_{1i}$

  $= \left((2.5 - 0.5) * 0.5\right) + 0.5$

  $= 1{,}5$

- $C_2 = \left(c_{2f} - c_{2i}\right)\frac{t}{t_{max}} + c_{2i}$

  $= (0.5 - 2.5) * 0.5) + 2.5$

  $= 1{,}5$

Perhitungan *update* kecepatan dapat dilakukan setelah mendapatkan nilai C_1, C_2, dan K. Untuk nilai r_1 dan r_2 masing-masing menggunakan nilai sebesar 0.9342 dan 0.3963. Berikut contoh perhitungan *update* kecepatan:

- $v_{1,1} = K[0{,}7V_{ij} + C_1 \times r_1(PBest_j - X_{ij}) + C_2 \times r_2(GBest_j - X_{ij})], t \geq \frac{T_{max}}{2}$

  $= 0{,}607341832\big((0{,}7 * 0) + (1{,}5 * 0{,}9342 * (9{,}45187 - 9{,}45187)) + (1{,}5 * 0{,}3963 * (9{,}45187 - 9{,}45187))\big)$

  $= 0$

- $v_{2,1} = K[0{,}7V_{ij} + C_1 \times r_1(PBest_j - X_{ij}) + C_2 \times r_2(GBest_j - X_{ij})], t \geq \frac{T_{max}}{2}$

  $= 0{,}607341832\big((0{,}7 * 0) + (1.5 * 0{,}9342 * (6{,}40061 - -6{,}40061)) + (1{,}5 * 0{,}3963 * (9{,}45187 - 6{,}40061))\big)$

  $= 1{,}1016$

Berikut adalah hasil perhitungan *update* kecepatan pada iterasi ke-1 ditampilkan pada Tabel 4.16.

Tabel 4.16 Hasil *Update* Kecepatan Iterasi ke-1

V1	Nilai Dimensi					
	SB	PH	PN	S	L	C
$V_{1,1}$	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
$V_{2,1}$	1.1016	1.58039893	-1.473801	12.34247	-1.80062	-7.4283
	3.8647	-0.9859570	0.035130	-2.86959	-2.49946	1.6973
	-0.2962	2.26627244	-0.39004	4.054145	-0.39001	8.1812
	-7.0336	-1.5059089	2.347665	-5.44669	2.22608	-7.5839
	3.8069	0.18762376	0.036028	-0.40883	0.546703	4.9383
	2.8889	-0.1142344	-0.361014	-6.20330	-1.59909	-4.122

4.4.6 Menghitung *Update* Posisi

Setelah mendapatkan seluruh nilai kecepatan yang baru tiap partikelnya, langkah selanjutnya adalah melakukan *update* posisi. Perhitungan *update* posisi dilakukan sesuai pada Persamaan 2.38. Berikut contoh perhitungan *update* posisi:

- $X_1^{0+1} = X_1^0 + V_1^{0+1}$

  $= 9{,}45187 \pm 0$

  $= 9{,}45187$

- $X_2^{0+1} = X_2^0 + V_2^{0+1}$

$$= 6{,}40061 + 1{,}1016$$

$$= 7{,}50222$$

Posisi seluruh partikel dihitung berdasarkan posisi awal dan nilai kecepatan yang baru dan pada Tabel 4.17 menunjukkan hasil dari posisi terbaru seluruh partikel pada iterasi ke-1:

Tabel 4.17 Hasil *Update* Posisi Iterasi ke-1

Partikel	Nilai Dimensi					
	SB	PH	PN	S	L	C
X1	9.45186	8.5038696	1.500422	45.7776	3.934177	4.85250
	17.5731	1.0997178	8.452557	4.10774	2.983667	32.151
	17.0219	8.7403305	3.64618	25.9433	8.223065	31.2091
	1.07307	3.6381138	8.097785	8.56712	7.313755	10.4603
	20.0392	2.5687798	1.641909	22.8714	3.411255	30.6917
	14.3551	4.6928929	3.485251	8.93516	6.279726	16.0374
X2	7.50222	5.7068487	4.108785	23.9336	7.1209619	17.9993
	10.7333	2.8446837	8.390382	9.18640	7.4072670	29.1476
	17.5461	4.7294372	4.336497	18.7682	8.9133159	16.7297
	13.5213	6.3033008	3.942841	18.2067	3.3739803	23.8825
	13.3016	2.2367196	1.578146	23.5950	2.4436878	21.9518
	9.24219	4.8950674	4.124182	19.9138	9.1098415	23.3333

4.4.7 Menghitung *Fitness*

Langkah selanjutnya setelah melakukan *update* posisi adalah menghitung nilai *fitness* baru. Tahap-tahap perhitungan sama dengan perhitungan *fitness* sebelumnya. Berikut merupakan hasil perhitungan *fitness* pada iterasi ke-1 ditampilkan pada Tabel 4.18.

Tabel 4.18 Nilai *Fitness* Iterasi ke-1

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
X1	9.45186	8.5038696	1.500422	45.7776	3.934177	4.85250	77.317
	17.5731	1.0997178	8.452557	4.10774	2.983667	32.151	
	17.0219	8.7403305	3.64618	25.9433	8.223065	31.2091	
	1.07307	3.6381138	8.097785	8.56712	7.313755	10.4603	
	20.0392	2.5687798	1.641909	22.8714	3.411255	30.6917	
	14.3551	4.6928929	3.485251	8.93516	6.279726	16.0374	
X2	7.50178	5.7062195	4.10937	23.9287	7.1216789	18.0023	67.746
	10.7317	2.8450763	8.390368	9.18754	7.4082623	29.1469	
	17.5463	4.7285349	4.336652	18.7666	8.9134712	16.7264	
	13.5241	6.3039004	3.941906	18.2089	3.3730939	23.8855	
	13.3001	2.2366449	1.578131	23.5951	2.4434701	21.9498	

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
	9.24104	4.8951129	4.124325	19.9163	9.1104783	23.3349	

4.4.8 *Update PBest* dan *GBest*

Langkah selanjutnya adalah melakukan pembaharuan nilai *PBest* dan *GBest* dari iterasi sebelumnya. Untuk memperbaharui *PBest*, nilai *fitness* iterasi sebelumnya dengan yang terbaru dibandingkan. Berikut ditampilkan *PBest* iterasi sebelumnya pada Tabel 4.19 dan posisi serta nilai *fitness* pada partikel terbaru pada Tabel 4.20.

Tabel 4.19 *PBest* Iterasi ke-0

PBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
PBest 1	9.45187	13.50387	1.50042	45.7777	3.93418	4.8525	77.317
	17.5732	12.099718	8.45256	4.10774	2.98367	32.1517	
	17.0219	8.7403305	3.64618	25.9434	8.22307	33.2091	
	1.07308	3.6381138	8.09779	8.56712	7.31376	10.4603	
	20.0392	2.5687799	1.64191	22.8715	3.41126	30.6918	
	14.3552	12.692893	3.48525	8.93517	6.27973	16.0375	
PBest 2	6.40061	14.12645	5.58259	11.5912	8.92159	25.4277	66.781
	6.86858	3.8306408	8.35525	12.056	9.90673	27.4503	
	17.8424	2.4631648	4.72655	14.7141	9.30333	8.5484	
	20.5549	17.80921	1.59518	23.6535	1.14789	31.4664	
	9.49467	10.049096	1.54212	24.0039	1.89698	17.0135	
	6.35322	17.009302	4.4852	26.1172	10.7089	27.4557	

Tabel 4.20 Posisi dan Nilai *Fitness* Partikel Iterasi Ke-1

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
X1	9.45186	8.5038696	1.500422	45.7776	3.93417	4.852	77.317
	17.5731	1.0997178	8.452557	4.10774	2.98366	32.15	
	17.0219	8.7403305	3.64618	25.9433	8.22306	31.20	
	1.07307	3.6381138	8.097785	8.56712	7.31375	10.46	
	20.0392	2.5687798	1.641909	22.8714	3.41125	30.69	
	14.3551	4.6928929	3.485251	8.93516	6.27972	16.03	
X2	7.50178	5.7062195	4.10937	23.9287	7.12167	18.00	67.746
	10.7317	2.8450763	8.390368	9.18754	7.40826	29.14	
	17.5463	4.7285349	4.336652	18.7666	8.91347	16.72	
	13.5241	6.3039004	3.941906	18.2089	3.37309	23.88	

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
	13.3001	2.2366449	1.578131	23.5951	2.44347	21.94	
	9.24104	4.8951129	4.124325	19.9163	9.11047	23.33	

Setelah membandingkan nilai *fitness* antara *PBest* iterasi ke-0 dengan posisi partikel pada iterasi ke-1, maka hasilnya dapat dilihat pada Tabel 4.21.

Tabel 4.21 *PBest* Iterasi ke-1

PBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
PBest 1	9.45186	8.5038696	1.500422	45.7776	3.934177	4.8525	77.317
	17.5731	1.0997178	8.452557	4.10774	2.983667	32.151	
	17.0219	8.7403305	3.64618	25.9433	8.223065	31.2091	
	1.07307	3.6381138	8.097785	8.56712	7.313755	10.4603	
	20.0392	2.5687798	1.641909	22.8714	3.411255	30.6917	
	14.3551	4.6928929	3.485251	8.93516	6.279726	16.0374	
PBest 2	7.50178	5.7062195	4.10937	23.9287	7.1216789	18.0023	67.746
	10.7317	2.8450763	8.390368	9.18754	7.4082623	29.1469	
	17.5463	4.7285349	4.336652	18.7666	8.9134712	16.7264	
	13.5241	6.3039004	3.941906	18.2089	3.3730939	23.8855	
	13.3001	2.2366449	1.578131	23.5951	2.4434701	21.9498	
	9.24104	4.8951129	4.124325	19.9163	9.1104783	23.3349	

Dari tabel di atas, nilai *fitness PBest* 2 iterasi ke-0 memiliki nilai yang lebih rendah dibandingkan dengan nilai *fitness* pada partikel ke-2 di iterasi ke-1, oleh karena itu terdapat proses *update PBest* 2 di iterasi ke-1.

Setelah *PBest* telah diperbaharui, dilanjutkan dengan mencari nilai *GBest* dari seluruh populasi. Pada *GBest* iterasi ke-1 ini dipilih dari nilai *fitness* terbesar pada *PBest* iterasi ke-1. Berikut adalah hasil *update GBest* pada iterasi ke-1 seperti pada Tabel 4.22.

Tabel 4.22 *GBest* Iterasi Ke-1

GBest	Nilai Dimensi						Fitness
	SB	PH	PN	S	L	C	
GBest	9.4518	8.503869	1.50042	45.777	3.93417	4.8525	77.317
	17.573	1.099717	8.45255	4.1077	2.98366	32.151	
	17.021	8.740330	3.64618	25.943	8.22306	31.2091	
	1.0730	3.638113	8.09778	8.5671	7.31375	10.4603	
	20.039	2.568779	1.64190	22.871	3.41125	30.6917	
	14.355	4.692892	3.48525	8.9351	6.27972	16.0374	

4.4.9 Iterasi Ke-2

Lanjut pada iterasi kedua yang dimulai dengan proses *update* kecepatan hingga *update GBest* iterasi ke-2 dan dapat dilihat pada Tabel 4.23 – 4.26.

***Update* Kecepatan**

Tabel 4.23 Hasil *Update* Kecepatan Iterasi Ke-2

V1	Nilai Dimensi					
	SB	PH	PN	S	L	C
$V_{1,1}$	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
$V_{1,2}$	0.99121447	1.422033466	-1.326122	11.105523	-1.62019344	-6.68402
	3.4773179	-0.88715835	0.0316092	-2.582045	-2.24900265	1.527162
	-0.2665392	2.039178325	-0.350965	3.6478338	-0.35093059	7.361406
	-6.3288315	-1.35500797	2.1124142	-4.901014	2.003020792	-6.82403
	3.42546101	0.168822626	0.0324165	-0.368043	0.491920714	4.443369
	2.59946956	-0.10278751	-0.324840	-5.581828	-1.43885919	-3.70936

Update* Posisi dan Hitung *Fitness

Tabel 4.24 Hasil *Update* Posisi dan Hitung *Fitness* Iterasi ke-2

Partikel	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
X1	9.451867	8.50386969	1.50042	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.45257	4.107744	2.983667	32.15166	
	17.02194	8.74033051	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.63811382	8.09778	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.64190	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.48525	8.935167	6.279726	16.03745	
X2	8.493435	7.12888225	2.78266	35.03921	5.500768	11.31533	76.432
	14.21062	1.95752543	8.42199	6.604354	5.158264	30.6748	
	17.27964	6.76861561	3.98553	22.41607	8.562385	24.0911	
	7.192479	4.94829288	6.05525	13.30577	5.377001	17.05848	
	16.72708	2.40554228	1.61056	23.22698	2.935608	26.39519	
	11.84166	4.79227990	3.79934	14.33206	7.670982	19.62393	

Update PBest

Tabel 4.25 Hasil *Update PBest* Iterasi Ke-2

PBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
PBest 1	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166	
	17.02194	8.74033051	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.63811382	8.097785	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745	
PBest 2	8.493435	7.12888225	2.782663	35.03921	5.500768	11.31533	76.432
	14.21062	1.95752543	8.421991	6.604354	5.158264	30.6748	
	17.27964	6.76861561	3.985531	22.41607	8.562385	24.0911	
	7.192479	4.94829288	6.055255	13.30577	5.3770011	17.05848	
	16.72708	2.40554228	1.610562	23.22698	2.9356085	26.39519	
	11.84166	4.79227990	3.799341	14.33206	7.6709824	19.62393	

Update GBest

Tabel 4.26 Hasil *GBest*

GBest	Nilai Dimensi						*Fitness*
	SB	PH	PN	S	L	C	
GBest	9.451867	8.50386969	1.500422	45.77765	3.934177	4.852503	77.317
	17.57319	1.09971783	8.452557	4.107744	2.983667	32.15166	
	17.02194	8.74033051	3.64618	25.94335	8.223065	31.20912	
	1.073075	3.63811382	8.097785	8.567124	7.313755	10.46032	
	20.03923	2.56877986	1.641909	22.87146	3.411255	30.69178	
	14.35516	4.69289298	3.485251	8.935167	6.279726	16.03745	

4.4.10 Hasil Optimasi

Setelah mencapai batas iterasi maksimal, maka *stop condition* telah dipenuhi. Dari hasil *GBest* terakhir menunjukkan solusi berupa komposisi bahan makanan optimal dengan kembali melakukan konversi di tiap partikel *GBest* ke indeks bahan makanan. Berikut merupakan hasil optimasi komposisi bahan makanan pada Tabel 4.27.

Tabel 4.27 Hasil Optimasi Komposisi Bahan Makanan

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)	Kalori (kal)	Karbo (gr)	Protein (gr)	Lemak (gr)
1	Pagi	Mie Basah	100	37.5	65.625	15	1.5	0
		Udang Segar	35	8.75	18.75	0	1.75	1.25

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)	Kalori (kal)	Karbo (gr)	Protein (gr)	Lemak (gr)
		Tahu	100	25	20	2	1.5	0.75
		Selada Air	100	25	0	0	0	0
		Minyak Jagung	5	1.25	12.5	0	0	1.25
	Camilan	Nanas	75	1.275	0.85	0.204	0	0
	Siang	Tepung Sagu	40	70	306.25	70	7	0
		Ayam Tanpa Kulit	40	10	12.5	0	1.75	0.5
		Kacang Merah	15	9.375	50	5	3.75	1.875
		Daun Pepaya	100	25	12.5	2.5	0.75	0
		Minyak Biji Bunga Matahari	5	1.25	12.5	0	0	1.25
	Camilan	Susu Kerbau	100	5	7.5	0.5	0.35	0.5
	Malam	Tepung Hungkwe	40	85	371.875	85	8.5	0
		Udang Segar	35	8.75	18.75	0	1.75	1.25
		Kacang Hijau	15	1.875	10	1	0.75	0.375
		Daun Pakis	100	25	6.25	1.25	0.25	0
		Mentega	15	3.75	12.5	0	0	1.25
	Camilan	Susu Kedelai	200	20	12.5	1	0.7	0.6
2	Pagi	Nasi	100	12.5	21.875	5	0.5	0
		Ikan Kering	15	3.75	12.5	0	1.75	0.5
		Kacang Merah	15	9.375	50	5	3.75	1.875
		Daun Talas	100	25	12.5	2.5	0.75	0
		Minyak Zaitun	5	1.25	12.5	0	0	1.25
	Camilan	Jambu Air	110	22	10	2.4	0	0
	Siang	Tepung Terigu	50	100	350	80	8	0
		Ikan Segar	50	12.5	12.5	0	1.75	0.5
		Tahu	100	25	20	2	1.5	0.75
		Tauge Kacang Hijau	100	25	6.25	1.25	0.25	0
		Minyak Biji Bunga Matahari	5	1.25	12.5	0	0	1.25
	Camilan	Susu Kedelai	200	20	12.5	1	0.7	0.6
	Malam	Krakers	50	62.5	218.75	50	5	0
		Bakso	100	250	187.5	0	17.5	12.5
		Oncom	50	25	40	4	3	1.5
		Daun Talas	100	25	12.5	2.5	0.75	0
		Minyak Kacang Tanah	5	1.25	12.5	0	0	1.25
	Camilan	Blewah	100	10	5	1.2	0	0

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (gr)	Total Berat (gr)	Kalori (kal)	Karbo (gr)	Protein (gr)	Lemak (gr)
Jumlah Nilai Gizi					1960.225	340.304	75.5	32.825
Rata-rata Nilai Gizi					980.1125	170.152	37.75	16.4125

4.5 Perancangan Pengujian

Dalam sub bab ini dijelaskan mengenai pengujian yang dilakukan dalam penelitian. Tujuan pengujian ini adalah untuk mendapatkan paremeter terbaik agar menghasilkan hasil yang paling optimal. Di akhir pengujian dilakukan analisis global untuk menguji kembali apakah hasil yang dikeluarkan sudah mencapai tujuan penelitian.

4.5.1 Perancangan Pengujian Parameter

Perancangan Pengujian Jumlah Populasi *(PopSize)*

Pengujian jumlah populasi ini bertujuan untuk mendapatkan jumlah *popSize* yang paling optimal untuk menunjang kinerja algoritme *Improved*-PSO. Pengujian *popSize* ini dilakukan sebanyak 10 kali pengujian dengan jumlah *popSize* yang digunakan mulai dari 10 hingga 100. Dengan 10 pengujian tersebut didapatkan nilai *fitness* yang berbeda-beda dari inisialisasi partikel yang berbeda pula. Setelah selesai melakukan 10 pengujian tersebut dihitung rata-rata dari seluruh *fitness* yang ada. *Popsize* dengan nilai rata-rata *fitness* tertinggi selanjutnya dipilih dan digunakan pada pengujian selanjutnya. Berikut rancangan pengujian *popSize* seperti yang tertera pada Tabel 4.28.

Tabel 4.28 Perancangan Pengujian Jumlah Populasi (*PopSize*)

popSize	*Fitness* percobaan ke-					Rata-rata *fitness*
	1	2	3	…	10	
10						
20						
30						
40						
50						
60						
70						
80						
90						
100						

Perancangan Pengujian Koefisien Akselerasi.

Pengujian koefisien akselerasi juga dilakukan sebanyak 10 kali pada masing-masing nilai C_1 dan C_2. Jumlah *popSize* yang digunakan untuk pengujian ini diambil *popSize* terbaik dari pengujian sebelumnya. Nilai C_1 dan C_2 yang digunakan dalam pengujian beragam. Berikut rancangan pengujian koefisien akselerasi seperti yang tertera pada Tabel 4.29.

Tabel 4.29 Perancangan Pengujian Koefisien Akselerasi.

Nilai C_1 dan C_2	*Fitness* percobaan ke-					Rata-rata *fitness*
	1	2	3	…	10	

4.5.2 Perancangan Pengujian Konvergensi

Pengujian konvergensi dilakukan untuk mengetahui pada iterasi berapa algoritme *Improved*-PSO mencapai titik konvergen. Pada pengujian konvergensi ini dilakukan sebanyak 5 kali dengan jumlah iterasi sebanyak 1000 iterasi. Pengujian konvergensi ini ditampilkan dalam bentuk grafik yang berisi nilai-nilai *fitness* yang beragam dari tiap iterasi dan percobaannya.

4.5.3 Perancangan Analisis Global

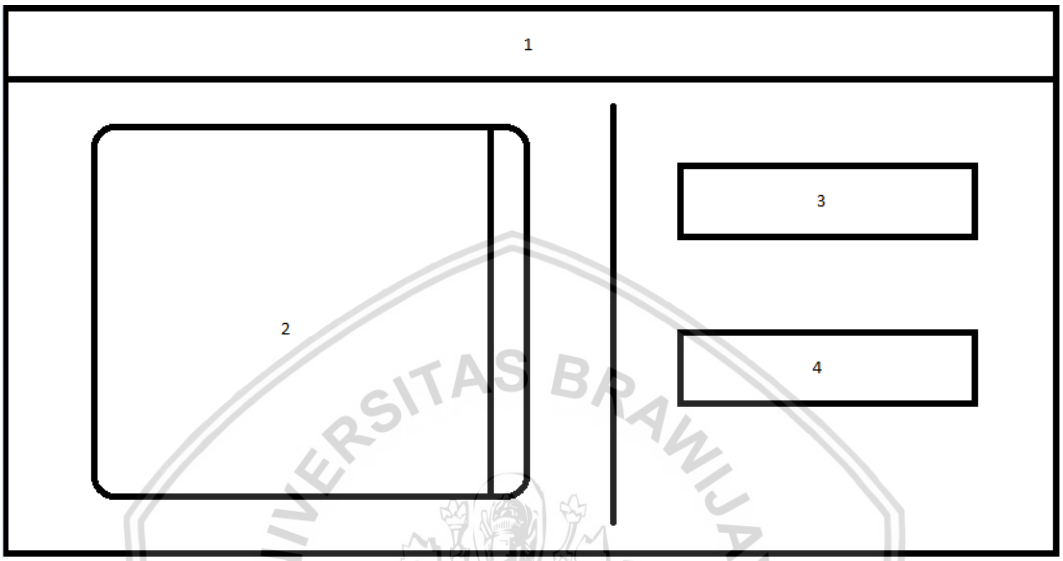
Analisi global dilakukan dengan tujuan untuk membandingkan kinerja sistem yang mengaplikasikan algoritme *Improved*-PSO dengan hasil asli dari pakar. Parameter yang digunakan dalam pengujian ini merupakan parameter terbaik dari keseluruhan pengujian sebelumnya.

4.6 Perancangan Antarmuka

Pada sub bab ini berisi mengenai rancangan antarmuka yang nantinya ditampilkan oleh sistem. Dalam rancangan antarmuka ini, penulis membuat 4 antarmuka berupa halaman utama, halaman *input*, halaman *output*, dan halaman daftar bahan makanan penukar.

4.6.1 Antarmuka Halaman Utama

Pada halaman utama ini, sistem menampilkan seluruh menu yang dapat diakses oleh pengguna. Menu tersebut antara lain daftar bahan makanan penukar, daftar data pasien, menu *input* data, dan deskripsi umum penelitian dan sistem. Berikut merupakan rancangan antarmuka halaman utama yang ditunjukkan pada Gambar 4.9.



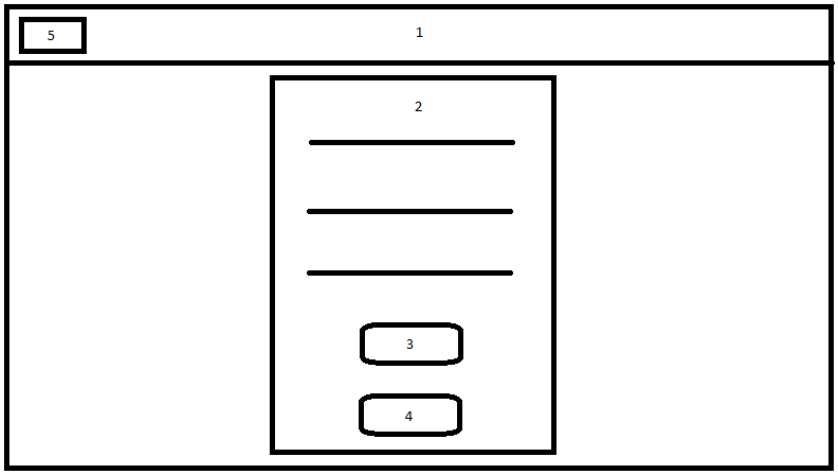
Gambar 4.9 Halaman Utama

Keterangan:

1. Judul Sistem
2. Deskripsi Umum Sistem
3. Menu *Input*
4. Menu Daftar Bahan Makanan Penukar

4.6.2 Antarmuka Halaman *Input*

Pada halaman *input* ini terdapat *form* yang disediakan oleh sistem yang tujuannya adalah agar pengguna memasukkan nilai parameter *Improved*-PSO yang digunakan untuk melakukan proses perhitungan komposisi makanan. Berikut merupakan rancangan antarmuka halaman *input* yang ditunjukkan pada Gambar 4.10.



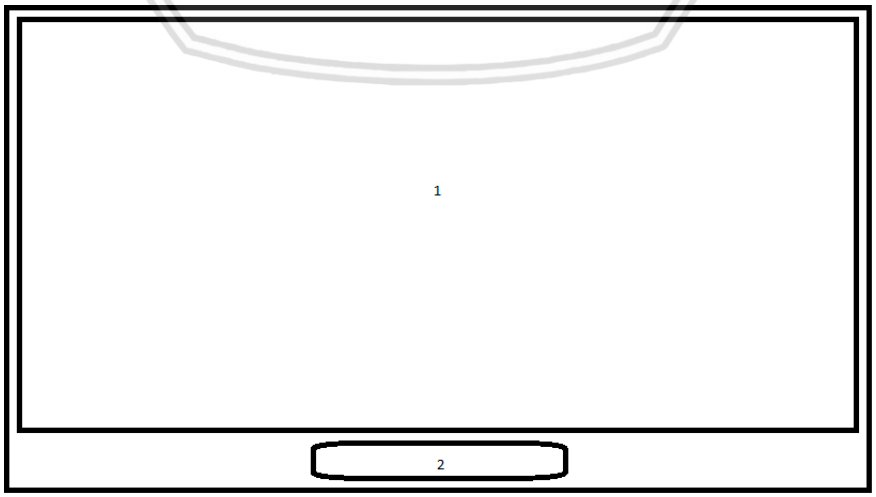
Gambar 4.10 Halaman *Input*

Keterangan:

1. Judul Sistem
2. *Form Input* Data
3. Tombol *OK* untuk melanjutkan proses
4. Tombol *Cancel* untuk membatalkan proses
5. Tombol *Back* untuk kembali ke halaman utama

4.6.3 Antarmuka Halaman *Output*

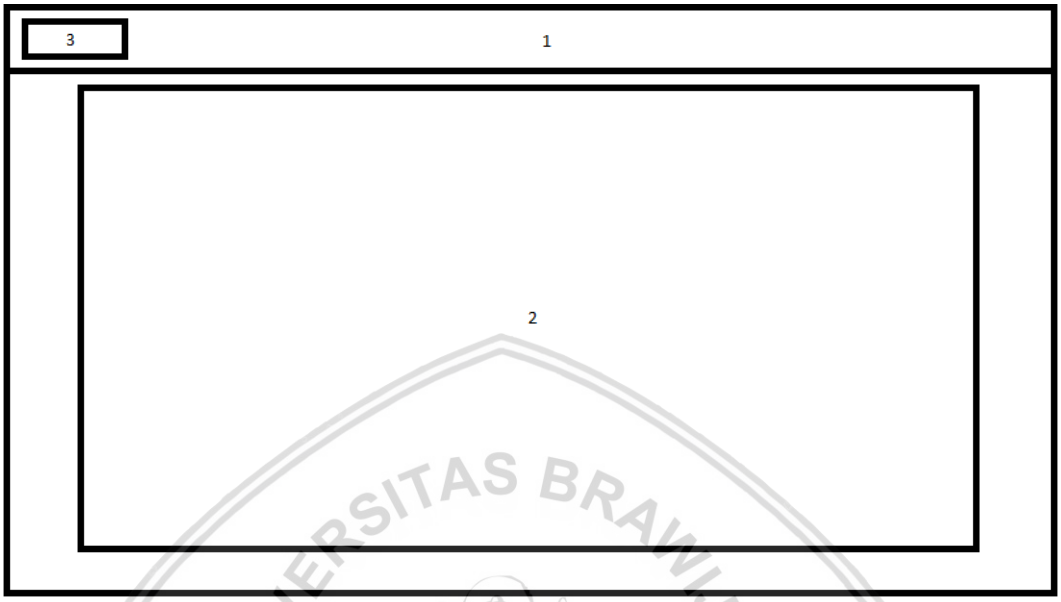
Pada halaman *ouput* ini dibagi menjadi dua bagian. Bagian pertama memperlihatkan proses serta hasil perhitungan sistem berupa nilai *GBest* seluruh iterasi. Bagian kedua adalah hasil akhir sistem berupa solusi dalam bentuk komposisi makanan yang berupa detail nama makanan hingga harga makanan. Berikut merupakan rancangan antarmuka halaman *output* yang ditunjukkan pada Gambar 4.11 dan 4.12.



Gambar 4.11 Halaman *Output* (I)

Keterangan:

1. Proses dan hasil perhitungan algoritme *Improved*-PSO
2. Tombol untuk melihat detail hasil rekomendasi komposisi bahan makanan



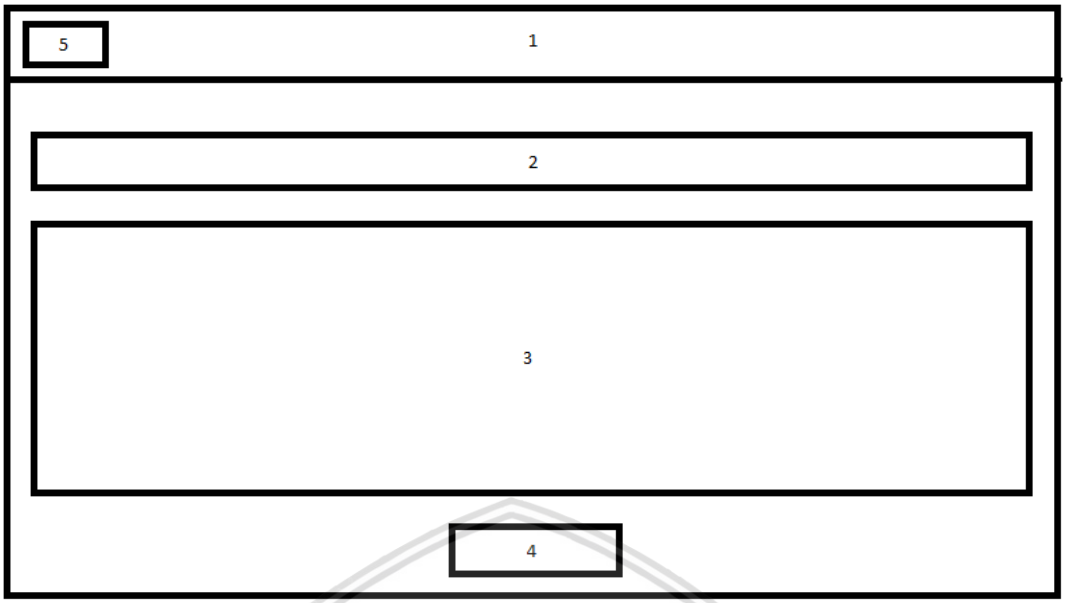
Gambar 4.12 Halaman *Output* (II)

Keterangan:

1. Judul Sistem
2. Detail Komposisi Bahan Makanan Hasil Rekomendasi Sistem
3. Tombol kembali ke menu *input*

4.6.4 Antarmuka Halaman Daftar Bahan Makanan Penukar

Pada halaman daftar bahan makanan penukar ini menyediakan data lengkap bahan makanan yang digunakan dalam penelitian. Pada halaman ini juga terdapat menu untuk melakukan perubahan daftar bahan makanan penukar yang dapat dilakukan oleh pengguna. Berikut merupakan rancangan antarmuka halaman *output* yang ditunjukkan pada Gambar 4.13.



Gambar 4.13 Halaman Daftar Bahan Makanan Penukar

Keterangan:

1. Judul Sistem
2. Pilihan Golongan Bahan Makanan (I-VII)
3. Detail Bahan Makanan Penukar
4. Tombol *Edit* untuk mengubah data bahan makanan penukar
5. Tombol *Back* untuk kembali ke halaman utama

BAB 5 IMPLEMENTASI

Bab ini berisi penjelasan mengenai implentasi dari sistem yang dirancang berdasarkan analisis kebutuhan yang telah dijabarkan sebelumnya. Implementasi dari algoritme *Improved*-PSO serta antarmuka yang digunakan dalam penyeleseaian masalah optimasi bahan makanan untuk pemenuhan gizi penderita diabetes juga dijelaskan dalam bab ini.

5.1 Spesifikasi Sistem

Spesifikasi sistem merupakan perangkat yang digunakan dalam melakukan implementasi dan membangun sistem.

5.1.1 Spesifikasi Perangkat Keras

Berikut merupakan spesifikasi perangkat keras yang digunakan dalam pengembangan sistem Implementasi algoritme *Improved*-PSO Untuk Optimasi Bahan Makanan Untuk Kebutuhan Gizi Penderita Diabetes Melitus:

- Processor Intel® Core™ i5-5200U CPU @ 2.20GHz 2.19 GHz
- Kapasitas Memori (RAM) sebesar 4.00 GB

5.1.2 Spesifikasi Perangkat Lunak

Berikut merupakan spesifikasi perangkat lunak yang digunakan dalam pengembangan sistem Implementasi algoritme *Improved*-PSO Untuk Optimasi Bahan Makanan Untuk Kebutuhan Gizi Penderita Diabetes Melitus:

- OS Windows 10 Proffesional 64 bit
- Bahasa Pemrograman HTML dan PHP versi: 7.0.2
- XAMPP v3.2.2
- HeidiSQL 9.5.0.5196
- Sublime Text Editor v3
- Materialize (Front-end framework)
- Google Chrome versi: 63.0.3239.132 64-bit

5.2 Implementasi Program

Sub-bab implementasi program berisi mengenai alur program optimasi bahan makanan menggunakan algoritme *Improved*-PSO mulai dari inisialisasi awal partikel hingga *decode* hasil akhir partikel optimal menjadi daftar bahan makanan yang aktual. Data yang dimasukkan berupa data diri pasien dan data yang dihasilkan berupa tabel bahan makanan yang optimal bagi pasien.

5.2.1 Proses *Input*

Pada proses *input* data pasien ini akan dioperasikan oleh pengguna *user* untuk memasukkan data berupa data diri pasien dan juga variabel-variabel utama dalam algoritme *Improved*-PSO seperti jumlah iterasi, jumlah populasi, dan jumlah hari untuk bahan makanan yang akan dikonsumsi pasien. Proses *input* ditunjukkan pada Kode Program 5.1.

```
<form action="index.php" method="POST" id="formDataDiri">
<center>
  <div class="row">
    <div class="input-field col s9" style="margin-
     left:50px;
     margin-right:-120px;">
     <i class="material-icons prefix">person</i>
     <input id="nama" name="nama" type="text"
      class="validate" required>
     <label style='text-align: left;' for="nama">Nama
      Pasien</label><center></div>
    <div class="input-field col s9" style="margin-
     left:50px;">
     <i class="material-icons prefix">accessibility</i>
     <input id="tinggi" name="tinggi" type="number"
      class="validate" required>
     <label style='text-align: left;' for="tinggi">Tinggi
      Badan Pasien</label><center>
    </div>
  </div>
  <div class="row">
    <div class="input-field col s9" style="margin-
     left:50px;
     margin-right:-120px;">
     <i class="material-icons prefix">exposure</i>
     <input id="berat" name="berat" type="number"
      class="validate" required>
     <label style='text-align: left;' for="berat">Berat
      Badan Pasien</label><center>
    </div>
    <div class="input-field col s9" style="margin-
     left:50px;">
     <i class="material-icons prefix">cake</i>
     <input id="umur" name="umur" type="number"
      class="validate" required>
     <label style='text-align: left;' for="umur">Usia
      Pasien</label><center>
    </div>
  </div>
  <div class="row">
    <div class="input-field col s9" style="margin-
     left:50px;
     margin-right:-120px;">
     <i class="material-icons prefix">work</i>
     <select name="Pekerjaan">
       <option value="" disabled selected>Pilih
        Aktivitas</option>
       <option value=”BedRest”>Bed Rest</option>
       <option value="Istirahat">Istirahat</option>
       <option value="Ringan">Aktivitas Ringan</option>
```

```
52          <option value="Sedang">Aktivitas Sedang</option>
53          <option value="Berat">Aktivitas Berat</option>
54          <option value="SBerat">Aktivitas Sangat
55           Berat</option>
56        </select>
57        <label style='text-align: left;'>Pilih
58         Pekerjaan</label><center>
59      </div>
60      <div class="input-field col s9" style="margin-
61       left:50px;">
62       <i class="material-icons prefix">wc</i>
63       <input id="laki" name="kelamin" type="radio"
64        value="Pria" checked>
65       <label style='text-align: left;'
66        for="laki">Pria</label>
67       <input id="perempuan" name="kelamin" type="radio"
68        value="Wanita">
69       <label style='text-align: left;'
70        for="perempuan">Wanita</label>
71      </div>
72    </div>
73    <div class="row">
74      <div class="input-field col s9" style="margin-
75       left:50px;
76       margin-right:-120px;">
77       <i class="material-icons prefix">grain</i>
78       <input id="popSize" name="popSize" type="number"
79        class="validate" required>
80       <label style='text-align: left;' for="umur">Jumlah
81        Populasi</label><center>
82      </div>
83      <div class="input-field col s9" style="margin-
84       left:50px;">
85       <i class="material-icons prefix">date_range</i>
86       <input id="jumlahHari" name="jumlahHari" type="number"
87        class="validate" required>
88       <label style='text-align: left;' for="umur">Jumlah
89        Hari</label><center>
90      </div>
91    </div>
92    <div class="row">
93      <div class="input-field col s9" style="margin-
94       left:50px;
95       margin-right:-120px;">
96       <i class="material-icons prefix">autorenew</i>
97       <input id="iterasiMaksimal" name="iterasiMaksimal"
98        type="number" class="validate" required>
99       <label style='text-align: left;' for="umur">Jumlah
100       Iterasi</label><center></div>
101   </div>
102   <div class="row">
103     <button class="col s8 btn waves-effect waves-light btn-
104      large z-depth-1 y-depth-1" type="submit" name="action"
105      style="margin-left:80px;">Hitung</button>
106   </div>
107 </form>
```

Kode Program 5.1 Implementasi Proses *Input*

Penjelasan Kode Program 5.1:

- Baris 1: Untuk mengirim hasil *input* data yang akan diproses dalam program.
- Baris 3-107: *Form* yang menjadi tempat untuk mengisi data pasien mulai dari nama hingga jenis kelamin dan juga 3 parameter algoritme *Improved*-PSO berupa jumlah populasi, jumlah iterasi, serta menentukan jumlah hari yang digunakan untuk optimasi.

5.2.2 Proses Hitung Kebutuhan Gizi Pasien

Proses kebutuhan gizi pasien dilakukan tepat setelah melakukan *input* data oleh *user* dan sebelum memasuki perhitungan algoritme *Improved*-PSO. Perhitungan kebutuhan gizi ini menggunakan data diri pasien untuk prosesnya. Proses hitung kebutuhan gizi pasien ditunjukkan pada Kode Program 5.2.

```
1   If (isset($_POST['nama'])) {
2      echo
3   "<script>document.getElementById('formDataDiri').style.display
4   = 'none';</script>";
5      $nama = $_POST['nama'];
6      $TB = $_POST['tinggi'];
7      $TB2 = $TB/100;
8      $BB = $_POST['berat'];
9      $U = $_POST['umur'];
10     $JK = $_POST['kelamin'];
11     $pekerjaan = $_POST['pekerjaan'];
12     $BBI = bbi($TB,$JK);
13     $IMT = $BB/($TB2*$TB2);
14     $kat = imt($TB,$BB);
15     $KKBJK = kkbjk($JK,$BB);
16     $KKBU = kkbu($U,$KKBJK);
17     $KKBA = kkba($KKBJK,$pekerjaan);
18     $KKBB = kkbb($KKBJK,$BB,$BBI);
19     $TotalKalori = $KKBJK-$KKBU+$KKBA+$KKBB;
20     $Karbo = 65/100*$TotalKalori/4;
21     $Lemak = 15/100*$TotalKalori/4;
22     $Protein = 20/100*$TotalKalori/9;
23  }
24  function bbi($TB,$JK){
25     if($TB<160&&$JK=="Pria"){
26        return $TB-100;
27     }
28     else if($TB<150&&$JK=="Wanita"){
29        return $TB-100;
30     } else {
31        return 0.9*($TB-100);
32     }
33        return 0;
34  }
35  function imt($TB,$BB){
36     $TB=$TB/100;
37     $IMT = $BB/($TB*$TB);
38     if($IMT<18.5){
39        return "Kurang";
40     }
```

```
else if($IMT>=18.5&&$IMT<=22.9){
      return "Normal";
   }
   else if($IMT>=23&&$IMT<=24.9){
      return "Dengan Risiko";
   }
   else if($IMT>=5&&$IMT<=29.9){
      return "Obes I";
   }
   else if($IMT>=30){
      return "Obes II";
   }
      return 0;
}
function kkbjk($JK, $BB){
   if($JK=="Pria"){
      return 30*$BB;
   }
   else if($JK=="Wanita"){
      return 25*$BB;
   }
      return 0;
}
function kkbu($U,$KKBJK){
   if($U<=59){
      return 5/100*$KKBJK;
   }
   else if($U>=60&&$U<=69){
      return 10/100*$KKBJK;
   }
   else if($U>=70){
      return 20/100*$KKBJK;
   }
   return 0;
}
function kkba($KKBJK, $pekerjaan){
   if($pekerjaan==”Bed Rest”){
      return 5/100*KKBJK;
   }
   else if($pekerjaan=="Istirahat"){
      return 10/100*$KKBJK;
   }
   else if($pekerjaan=="Ringan"){
      return 20/100*$KKBJK;
   }
   else if($pekerjaan=="Sedang"){
      return 30/100*$KKBJK;
   }
   else if($pekerjaan=="Berat"){
      return 40/100*$KKBJK;
   }
   else if($pekerjaan=="SBerat"){
      return 50/100*$KKBJK;
   }
   return 0;
}
function kkbb($KKBJK,$kat){
   $sg = ($BB/$BBI)*100;
   if($sg>=60&&$sg<=70){
```

```
99          return 30/100*$KKBJK;
100       }
101       else if($sg>70&&$sg<=80){
102           return 20/100*$KKBJK;
103       }
104       else if($sg>80&&$sg<=90){
105           return 10/100*$KKBJK;
106       }
107       else if($sg>=120&&$sg<=130){
108           return -(10/100*$KKBJK);
109       }
110       else if($sg>130&&$sg<=140){
111           return -(10/100*$KKBJK);
112       }
113       else if($sg>140){
114           return -(30/100*$KKBJK);
115       }
116       return 0;
117   }
```

Kode Program 5.2 Implementasi Proses Hitung Kebutuhan Gizi Pasien

Penjelasan Kode Program 5.2:

- Baris 1-4: Perhitungan dilakukan dengan mendapatkan data dari *form* yang diisi di awal program.
- Baris 5-11: Memanggil data yang dimasukkan di awal program.
- Baris 12-18: Melakukan perhitungan berat badan ideal, IMT, dan kebutuhan kalori basal dengan menggunakan fungsi.
- Baris 20-22: Menghitung total kebutuhan kalori berdasarkan kebutuhan kalori basal, serta menghitung total kebutuhan gizi berupa karbohidrat, protein, dan lemak.
- Baris 24-34: Menghitung berat badan ideal.
- Baris 35-54: Menghitung Indeks Massa Tubuh untuk menentukan kategori berat tubuh.
- Baris 55-63: Menghitung kebutuhan kalori basal berdasarkan jenis kelamin.
- Baris 64-75: Menghitung kebutuhan kalori basal berdasarkan usia.
- Baris 76-95: Menghitung kebutuhan kalori basal berdasarkan aktivitas fisik/pekerjaan.
- Baris 96-117: Menghitung kebutuhan kalori basal berdasarkan kategori berat tubuh.

5.2.3 Proses Inisialisasi Awal Partikel

Pada proses inisialisasi awal partikel ini membangkitkan nilai partikel dengan metode *Real-Code* PSO (RCPSO) sesuai dengan jumlah populasi dan jumlah hari yang diinginkan *user*. Proses inisialisasi awal partikel ditunjukkan pada Kode Program 5.3.

```
1  for ($i=0;$i<=$Tmax;$i++){
2    if ($i == 0){
3      for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
4        for ($k = 0; $k<6; $k++){
5            $random = rand(0,10000);
6            $X[$j][$k] = round($batasBawah + $random/10000 *
7            ($batasAtas[$k]-$batasBawah));
8        }
9    }
10 }
```

Kode Program 5.3 Implementasi Proses Inisialisasi Awal Partikel

Penjelasan Kode Program 5.3:

- Baris 1: Proses perulangan sesuai jumlah iterasi maksimal.
- Baris 2: Kondisi ketika iterasi=0.
- Baris 3-7: Proses inisialisasi awal partikel.

5.2.4 Proses Inisialisasi Kecepatan Awal Partikel

Proses inisialisasi kecepatan awal partikel menghasilkan nilai 0 karena masih belum mengalami perbaruan kecepatan pada iterasi ke-0. Proses inisialisasi kecepatan awal partikel ditunjukkan pada Kode Program 5.3.

```
1  for ($i=0;$i<=$Tmax;$i++){
2    if ($i == 0){
3      for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
4        for ($k = 0; $k<6; $k++){
5            $V[$j][$k] = 0;
6        }
7    }
8  }
```

Kode Program 5.4 Implementasi Proses Inisialisasi Kecepatan Awal Partikel

Penjelasan Kode Program 5.4:

- Baris 1: Proses perulangan sesuai jumlah iterasi maksimal.
- Baris 2: Kondisi ketika iterasi=0.
- Baris 3-5: Proses inisialisasi kecepatan awal partikel.

5.2.5 Proses Hitung *Fitness*

Proses hitung *fitness* menunjukkan bagaimana alur perhitungan *fitness* dari partikel yang ada di tiap populasinya. Mulai dari perhitungan kandungan gizi tiap makanan berupa perubahan berat, perhitungan kandungan kalori, karbohidrat, protein, dan lemak hingga perhitungan penalti gizi bahan makanan serta jumlah variasinya. Proses hitung *fitness* ditunjukkan pada Kode Program 5.5.

```
for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
  for ($k = 0; $k<6; $k++){
    $sql = "SELECT * FROM bahankandungan WHERE
    nourut=".round($X[$j][$k])." AND
    golmakanan='".findGolongan($k)."'";
    $val = mysqli_fetch_array(mysqli_query($con,$sql));
    $P[$j][$k][0] = 0;
    for ($l = 0; $l<6; $l++){
      $P[$j][$k][$l]=calP($l, $k, $val[4], $P[$j][$k][0],
      $val[3], $val[7], $val[8], $val[9], $val[10],
      calVariasi($j, $k, $jumlahHari, $popSize,
      round($X[$j][$k]), $X));
    }
    $totalSemua[0] += $P[$j][$k][1];
    $totalSemua[1] += $P[$j][$k][2];
    $totalSemua[2] += $P[$j][$k][3];
    $totalSemua[3] += $P[$j][$k][4];
    $totalSemua[4] += $P[$j][$k][5];
  }
  if (($j+1)%$jum==0){
     $penaltiAkhir[0] = abs($TotalKalori-
       ($totalSemua[0]/$jumlahHari));
     $penaltiAkhir[1] = abs($Karbo-
       ($totalSemua[1]/$jumlahHari));
     $penaltiAkhir[2] = abs($Protein-
       ($totalSemua[2]/$jumlahHari));
     $penaltiAkhir[3] = abs($Lemak-
       ($totalSemua[3]/$jumlahHari));

     $penaltiSemua = $penaltiAkhir[0] + $penaltiAkhir[1] +
       $penaltiAkhir[2] + $penaltiAkhir[3];
     $totalSemua[0] = 0;
     $totalSemua[1] = 0;
     $totalSemua[2] = 0;
     $totalSemua[3] = 0;
     $fitnessAkhir[$c-1] = ((1/$penaltiSemua)*100000) +
       $totalSemua[4];
     $totalSemua[4] = 0;
   }
}
function calVariasi($j, $k, $jumlahHari, $popSize, $data,
$array){
   $start = floor($j/($jumlahHari*3))*($jumlahHari*3);
   $end = (floor($j/($jumlahHari*3))+1)*($jumlahHari*3);
   for ($i = $start; $i < $end; $i++){
       if ($data == round($array[$i][$k]) && $j != $i){
           return 0;
       }
   }
   return 1;
}
function calP($i, $j, $urt, $beratUbah, $berat, $kalori,
$karbohidrat, $protein, $lemak, $var){
   switch ($i) {
      case 0:
           if ($j==5){
               return ($berat*$urt)*0.10;
           } else {
               return ($berat*$urt)*0.25;
```

```
60              }
61              break;
62          case 1:
63              return ($beratUbah/$berat)*$kalori;
64              break;
65          case 2:
66              return ($beratUbah/$berat)*$karbohidrat;
67              break;
68          case 3:
69              return ($beratUbah/$berat)*$protein;
70              break;
71          case 4:
72              return ($beratUbah/$berat)*$lemak;
73              break;
74          case 5:
75              return $var;
76              break;
77          default:
78              return 0;
79              break;
80      }
81  }
82  function findGolongan($i){
83      switch ($i) {
84          case 0:
85              return "KARBOHIDRAT";
86              break;
87          case 1:
88              return "HEWANI";
89              break;
90          case 2:
91              return "NABATI";
92              break;
93          case 3:
94              return "SAYURAN";
95              break;
96          case 4:
97              return "LEMAK";
98              break;
99          case 5:
100             return "CAMILAN";
101             break;
102         default:
103             return 0;
104             break;
105     }
106 }
```

Kode Program 5.5 Implementasi Proses Hitung *Fitness*

Penjelasan Kode Program 5.5:

- Baris 1: Melakukan perulangan sesuai dengan jumlah *popSize* dan jumlah hari untuk melakukan perhitungan tiap partikelnya.
- Baris 2: Melakukan perulangan sebanyak jenis bahan makanan yang ada.

- Baris 3-5: Melakukan koneksi ke *database* untuk mengkonversi nilai yang ada di tiap dimensinya untuk dipanggil sesuai indeks makanan dan sesuai dengan golongan makanannya.
- Baris 6: Menggunakan variabel *$val2* sebagai penghubung ke *database*.
- Baris 7: Inisialisasi variabel *$P* untuk menghitung kandungan gizi tiap bahan makanan untuk menghitung nilai penalti dan *fitness*.
- Baris 8-13: Perulangan sesuai jumlah dimensi untuk memanggil bahan makanan yang sesuai dengan indeks di inisialisasi awal. Sekaligus menghitung kandungan gizinya, serta menghitung jumlah variasi tiap partikel.
- Baris 14-18: Menghitung total kalori, karbohidrat, protein, dan lemak bahan makanan sesuai dengan kebutuhan dan menjumlah variasi dalam satu populasi.
- Baris 20-40: Proses menghitung penalti kalori, karbohidrat, protein, dan lemak dengan nilai yang absolut. Setelah mendapatkan 4 penalti tersebut, dilanjutkan dengan menghitung total penalti dan *fitness*.
- Baris 41-51: Mencari variasi bahan makanan di tiap partikelnya. Apabila terdapat bahan makanan yang muncul lebih dari satu kali, maka nilainya adalah 0 dan bila hanya muncul sekali nilainya adalah 1.
- Baris 52-81: Memanggil nilai-nilai kandungan bahan makanan mulai dari berat asli, berat yang telah dipangkas sesuai takaran porsi dan urt, kandungan karbohidrat, protein, dan lemak.
- Baris 82-106: Mencari golongan bahan makanan yang sesuai dengan posisi dimensi masing-masing nilai.

5.2.6 Proses Inisialisasi *PBest* & *GBest* Awal

Proses inisialisasi *PBest* dan *GBest* awal bertujuan untuk menampilkan *PBest* yang isinya sama dengan nilai partikel di tiap populasi, sedangkan *GBest* berisi populasi dengan nilai *fitness* yang terbesar. Proses inisialisasi *PBest* dan *GBest* awal ditunjukkan pada Kode Program 5.6.

```
1  $pBest = $X;
2  $pBestFitness = $fitnessAkhir;
3  $fitnessMax = $fitnessAkhir;
4  $max = max($fitnessAkhir);
5  for($m = 0; $m < count($popSize); $m++){
6    if($pBestFitness[$m] >= $max){
7       $start = $m*($jumlahHari*3);
8       $end = ($m+1)*($jumlahHari*3);
9       for ($q = $start; $q<$end; $q++){
10          $gBest[$q] = $pBest[$q];
11      }break;
12   }
13 }
```

Kode Program 5.6 Implementasi Proses Inisialisasi *PBest*&*GBest* Awal

Penjelasan Kode Program 5.6:

- Baris 1: Inisialisasi nilai *PBest* disesuaikan dengan nilai partikel yang sudah muncul.
- Baris 2: *PBest fitness* menyesuaikan dengan nilai *fitness* partikel.
- Baris 3: Mengisi variabel *fitnessMax* untuk mencari nilai *fitness* yang terbesar.
- Baris 4: Variabel *max* untuk mencari nilai *fitness* terbesar dalam *PBest*.
- Baris 5-13: Perulangan untuk mencari indeks *GBest* dan sekaligus mencari *fitness GBest*.

5.2.7 Proses *Update* Kecepatan

Proses *update* kecepatan menunjukkan perubahan kecepatan yang diawali oleh masing-masing partikel ketika mulai memasuki iterasi pertama hingga mencapai iterasi maksimal. Proses *update* kecepatan ditunjukkan pada Kode Program 5.7.

```
1   $c1f = 2.5;
2   $c1i = 0.5;
3   $c2f = 0.5;
4   $c2i = 2.5;
5   $c1 = ($c1f-$c1i)*$i/$Tmax+$c1i;
6   $c2 = ($c2f-$c2i)*$i/$Tmax+$c2i;
7   $acak1 = rand(0,10000);
8   $acak2 = rand(0,10000);
9   $r1 = $acak1/10000;
10  $r2 = $acak2/10000;
11  $inertia = 0.857143+((1-0.857143)*(1-($i/$Tmax)));
12  $K = (cos((2*3.14/$Tmax)*(($i-$Tmax)/2))+2.428571)/4;
13  $tt = $Tmax/2;
14  $nLoop = 0;
15  for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
16      if ($j%($jumlahHari*3)==0) $nLoop=0; else $nLoop++;
17      for ($k = 0; $k<6; $k++){
18          if($i<$tt){
19            if($X[$j][$k]==$gBest[$nLoop][$k]){
20              $w = 0.857143;
21            }
22            else
23            {
24              $w = $inertia;
25            }
26            $nilaiV = ($w *
27              $V[$j][$k])+($c1*$r1*($pBest[$j][$k]-
28              $X[$j][$k]))+($c2*$r2*($gBest[$nLoop][$k]-
29              $X[$j][$k]));
30            $V[$j][$k] = $nilaiV;
31          }
32          else
33          {
34            $nilaiV = $K * ((0.7 *
35              $V[$j][$k])+($c1*$r1*($pBest[$j][$k]-
36              $X[$j][$k]))+($c2*$r2*($gBest[$nLoop][$k]-
37              $X[$j][$k])));
```

```
38              $V[$j][$k] = $nilaiV;
39            }}}
```

Kode Program 5.7 Implementasi Proses *Update* Kecepatan

Penjelasan Kode Program 5.7:

- Baris 14: Variabel *nLoop* untuk mengambil partikel *GBest* yang sudah terbentuk.
- Baris 15-17: Perulangan serta kondisi untuk mengambil partikel *PBest* dan *GBest* dari iterasi sebelumnya yang digunakan untuk perhitungan *update* kecepatan.
- Baris 18-30: Kondisi saat *update* kecepatan terjadi pada jumlah iterasi kurang dari setengah iterasi maksimal.
- Baris 19-21: Kondisi saat partikel yang di*update* sama dengan partikel *GBest* iterasi sebelumnya.
- Baris 22-25: Kondisi saat partikel yang di*update* berbeda dengan partikel *GBest* iterasi sebelumnya.
- Baris 18-31: Rumus perhitungan *update* kecepatan dengan nilai *inertia weight*.
- Baris 32-39: Rumus perhitungan *update* kecepatan dengan *constriction factor*.

5.2.8 Proses Update Posisi

Proses *update* posisi ini menunjukkan perubahan posisi tiap partikel setelah adanya *update* kecepatan sehingga menghasilkan perubahan nilai *fitness* pada tiap partikel yang mengalami perubahan posisi/nilai. Proses *update* posisi ditunjukkan pada Kode Program 5.8.

```
1  for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
2      for ($k = 0; $k<6; $k++){
3        $nilai = $XTemp[$j][$k]+$V[$j][$k];
4        $XTemp[$j][$k] = $nilai;
5      }
6  }
```

Kode Program 5.8 Implementasi Proses *Update* Posisi

Penjelasan Kode Program 5.8:

- Baris 1: Perulangan sebanyak jumlah populasi.
- Baris 2: Perulangan sebanyak jumlah dimensi.
- Baris 3-4: Menghitung posisi yang baru dengan menambahkan nilai partikel yang lama dengan hasil dari *update* kecepatan. Nilai posisi yang baru disimpan dalam variabel *XTemp[]*.

5.2.9 Proses Menghitung Batas Maksimal dan Minimal Posisi

Proses menghitung batas maksimal dan minimal posisi ini digunakan agar perubahan posisi yang ada tidak melebihi batas atas dan bawah dari tiap partikel.

Proses menghitung batas maksimal dan minimal posisi ditunjukkan pada Kode Program 5.9.

```
1  for ($j = 0; $j<$popSize*$jumlahHari*3; $j++){
2      for ($k = 0; $k<6; $k++){
3        if($XTemp[$j][$k]>=$batasAtas[$k]){
4           $XTemp[$j][$k]=$batasAtas[$k];
5        }if($XTemp[$j][$k]<=$batasBawah){
6           $XTemp[$j][$k]=$batasBawah;}}
```

Kode Program 5.9 Implementasi Proses Menghitung Batas Maksimal dan Minimal Posisi

Penjelasan Kode Program 5.9:

- Baris 1: Perulangan sesuai jumlah populasi.
- Baris 2: Perulangan sesuai jumlah dimensi.
- Baris 3-4: Kondisi saat nilai posisi melebihi nilai batas bawah partikel.
- Baris 5-6: Kondisi saat nilai posisi melebihi nilai batas atas partikel.

5.2.10 Proses *Update PBest* & *GBest*

Proses *Update PBest* dan *GBest* digunakan untuk memperbaharui nilai *fitness* terbaik yang ada di tiap populasinya sebagai *PBest* dan juga nilai *fitness* terbaik di keseluruhan populasi pada semua iterasi sebagai *GBest*. Proses *update PBest* dan *GBest* ditunjukkan pada Kode Program 5.10.

```
1  for ($z = 0; $z < $popSize; $z++){
2      if ($fitnessMax[$z] <= $fitnessAkhir[$z]){
3          $fitnessMax[$z] = $fitnessAkhir[$z];}
4  }
5  for ($n = 0; $n < $popSize; $n++){
6      if($fitnessMax[$n] < $fitnessAkhirTemp[$n]){
7          $start = $n*($jumlahHari*3);
8          $end = ($n+1)*($jumlahHari*3);
9          for($q = $start; $q<$end; $q++){
10             $pBest[$q] = $XTemp[$q];}
11         $pBestFitness[$n] = $fitnessAkhirTemp[$n];
12     }
13 }
14 $X = $XTemp;
15 $fitnessAkhir = $fitnessAkhirTemp;
16 $max = max($pBestFitness);
17 for ($m = 0; $m < $popSize; $m++){
18     if($pBestFitness[$m] >= $max){
19         $start = $m*($jumlahHari*3);
20         $end = ($m+1)*($jumlahHari*3);
21         $cGBest = 0;
22         for ($q = $start; $q<$end; $q++){
23             $gBest[$cGBest++] = $pBest[$q];}
24     }
25 }
26 $fitnessFinal = max($pBestFitness);
```

Kode Program 5.10 Implementasi Proses Update PBest&GBest

Penjelasan Kode Program 5.10:

- Baris 1: Perulangan sebanyak jumlah *popsize*.
- Baris 2-4: Variabel *fitnessMax* digunakan untuk menyimpan nilai *fitness* pada iterasi sebelumnya.
- Baris 5: Perulangan sebanyak jumlah *popsize*.
- Baris 6: Kondisi yang digunakan untuk membandingkan nilai *fitness* iterasi sebelumnya dengan nilai *fitness* terbaru yang tersimpan dalam variabel *fitnessAkhirTemp[]*.
- Baris 9-11: Proses pembentukan nilai *PBest* yang baru.
- Baris 11: Inisialisasi nilai *PBestFitness* yang diambil dari *fitnessAkhirTemp* yang merupakan nilai *fitness* yang terbesar dari tiap populasi.
- Baris 14: Mengembalikan seluruh nilai *XTemp* menjadi nilai *X*.
- Baris 15: Mengembalikan seluruh nilai *FitnessAkhirTemp* menjadi nilai *fitnessAkhir*.
- Baris16: Inisialisasi variabel *max* untuk mengambil nilai *PBestFitness* yang terbesar.
- Baris 17: Perulangan sesuai dengan jumlah *popsize*.
- Baris 18: Kondisi untuk memilih *fitness* terbesar diantara nilai *fitness PBest*.
- Baris 19-25: Proses untuk pembentukan nilai *GBest*.
- Baris 26: Inisialisasi variabel *fitnessFinal* untuk mengambil 1 nilai *fitness PBest* yang terbesar.

5.2.11 Proses *Decode* Partikel

Proses *decode* partikel dilakukan pada saat program mencapai batas iterasi maksimal. Nilai partikel yang terdapat pada populasi yang mengandung nilai *GBest* dikonversikan ulang menjadi nama bahan makanan yang sesuai dengan indeksnya. Proses *decode* partikel ditunjukkan dalam Kode Program 5.11.

```
Session_start();
$pinalti = $penaltiSemua;
$fitnessGBest = $fitnessAkhirTempX[$c-1];
$_SESSION['nama'] = $nama;
$_SESSION['TB'] = $TB;
$_SESSION['BB'] = $BB;
$_SESSION['U'] = $U;
$_SESSION['JK'] = $JK;
$_SESSION['pekerjaan'] = $pekerjaan;
$_SESSION['kalori'] = $TotalKalori;
$_SESSION['karbo'] = $Karbo;
$_SESSION['protein'] = $Protein;
$_SESSION['lemak'] = $Lemak;
$_SESSION['hari'] = $jumlahHari;
$_SESSION['tes'] = $gBest;
$_SESSION['penalti'] = $pinalti;
$_SESSION['fitness'] = $fitnessGBest;
$_SESSION['kcal'] = $kcal;
$_SESSION['kkarbo'] = $kkarbo;
$_SESSION['kprotein'] = $kprotein;
$_SESSION['klemak'] = $klemak;
echo "<a href='tes.php'>Detail Makanan</a>";
-------------------------------------------------------------
session_start();
$gBest = $_SESSION['tes'];
$nama = $_SESSION['nama'];
$TB = $_SESSION['TB'];
$BB = $_SESSION['BB'];
$U = $_SESSION['U'];
$JK = $_SESSION['JK'];
$pekerjaan = $_SESSION['pekerjaan'];
$TotalKalori = $_SESSION['kalori'];
$Karbo = $_SESSION['karbo'];
$Protein = $_SESSION['protein'];
$Lemak = $_SESSION['lemak'];
$jumlahHari = $_SESSION['hari'];
$kcal = $_SESSION['kcal'];
$kkarbo = $_SESSION['kkarbo'];
$kprotein = $_SESSION['kprotein'];
$klemak = $_SESSION['klemak'];
$pinalti = $_SESSION['penalti'];
$fitness = $_SESSION['fitness'];
for($j = 0; $j<$xx*$jumlahHari*3; $j++){
  for($k = 0; $k<6; $k++){
     $sql = "SELECT * FROM bahankandungan WHERE
     nourut=".round($gBest[$j][$k])." AND
     golmakanan='".findGolongan($k)."'";
     $val = mysqli_fetch_array(mysqli_query($con,$sql));
  }
}
for($j = 0; $j<$xx*$jumlahHari*3; $j++){
  for($k = 0; $k<6; $k++){
     $sql = "SELECT * FROM bahankandungan WHERE
     nourut=".round($gBest[$j][$k])." AND
     golmakanan='".findGolongan($k)."'";
     $val = mysqli_query($con,$sql);
     while($row=mysqli_fetch_array($val)){
       $namamakanan=$row[2];
       $berat=$row[3];
```

```
60          if($k==5){
61              $beratptg=$row[3]*$row[4]*0.10;
62          } else {
63              $beratptg=$row[3]*$row[4]*0.25;
64          }
65          $klr = ($beratptg/$row[3])*$row[7];
66          $krb = ($beratptg/$row[3])*$row[8];
67          $prt = ($beratptg/$row[3])*$row[9];
68          $lmk = ($beratptg/$row[3])*$row[10];
69        }
70    }
71  }
```

Kode Program 5.11 Implementasi Proses Decode Partkel

Penjelasan Kode Program 5.11:

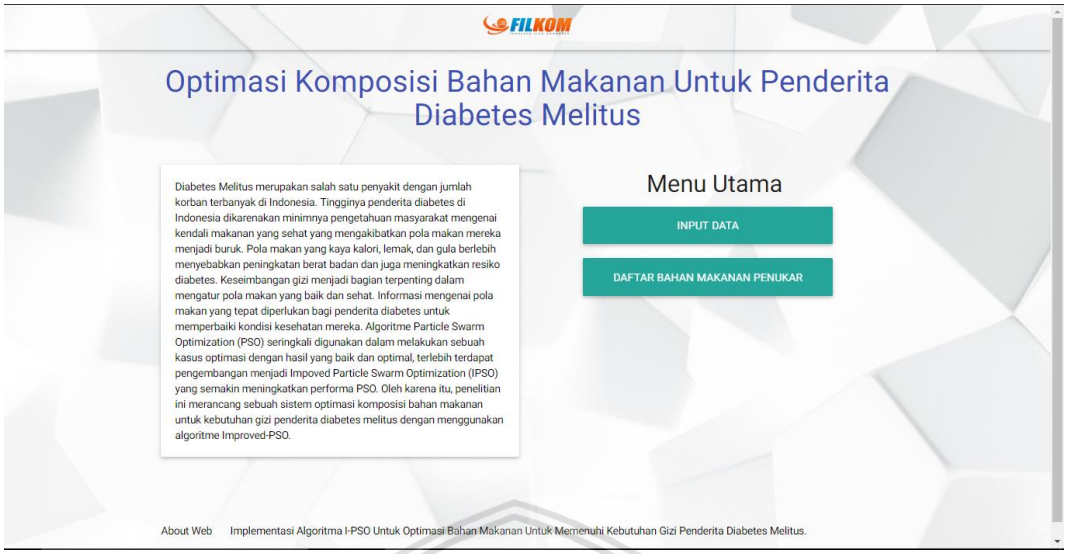
- Baris 1: Memulai *session*.
- Baris 1: Menyimpan nilai penalti dengan variabel *$penalti*.
- Baris 2: Menyimpan nilai fitness dengan variabel $*fitness*.
- Baris 3-20: Menyimpan data detail pasien dan perhitungan kebutuhan gizi harian pasien. Nilai penalti dan *fitness* juga disimpan.
- Baris 21: Mengirim data ke halaman *tes.php*.
- Baris 24: Memulai *session*.
- Baris 25-42: Memanggil data detail pasien dan perhitungan kebutuhan gizi harian pasien. Nilai penalti dan *fitness* juga dipanggil.
- Baris 43-50: Perulangan untuk menampilkan hasil *decode* bahan makanan yang paling optimal sesuai dengan indeks *GBest* akhir.
- Baris 51-71: Perulangan untuk menampilkan detail bahan makanan optimal sebagai hasil rekomendasi sistem.

5.3 Implementasi Antarmuka

Implementasi antarmuka sistem memiliki 4 antarmuka utama program yaitu halaman utama, halaman *input* data pasien, halaman *output*, dan halaman daftar bahan makanan penukar.

5.3.1 Implementasi Antarmuka Halaman Utama

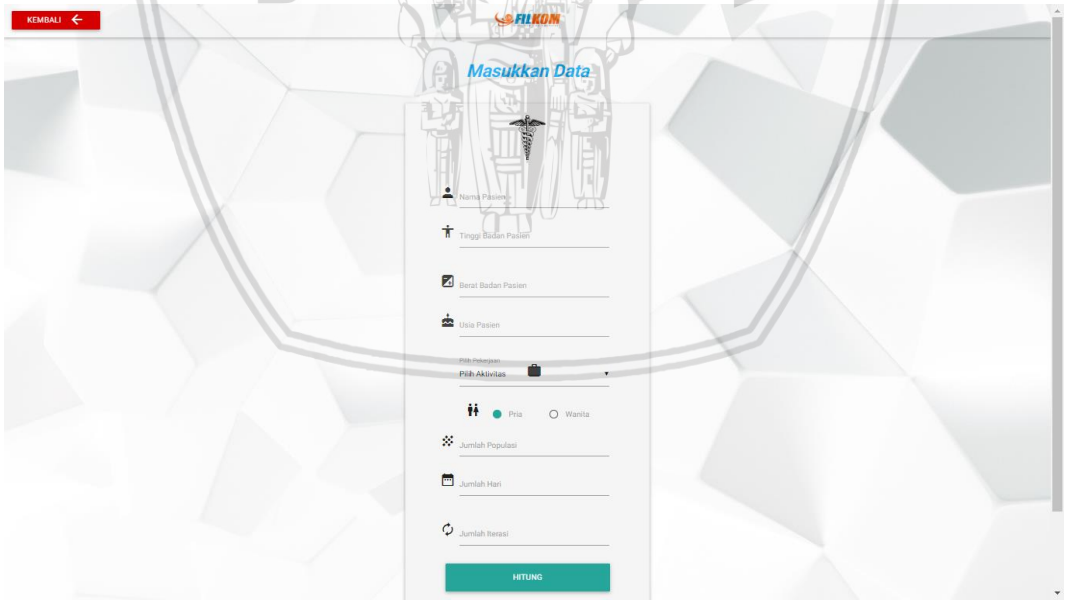
Pada implementasi antarmuka ini menampilkan halaman utama sistem yang terdiri dari judul penelitian, gambaran umum sistem, dan menu utama yang tersedia dalam sistem. Menu utama yang tersedia di sini berjumlah 2 menu yaitu *input* data dan daftar bahan makanan penukar. Implementasi antarmuka halaman utama ditunjukkan pada Gambar 5.1.



Gambar 5.1 Antarmuka Halaman Utama

5.3.2 Implementasi Antarmuka Halaman *Input*

Pada halaman *input* data ini berisi *form* yang dapat diisi dengan data diri pasien dan kondisi tubuhnya dan juga *form* untuk mengisi variabel dasar yang dibutuhkan dalam perhitungan algoritme *Improved*-PSO seperti jumlah populasi, jumlah iterasi, dan jumlah hari. Implementasi antarmuka halaman *input* ditunjukkan pada Gambar 5.2.



Gambar 5.2 Antarmuka Halaman *Input*

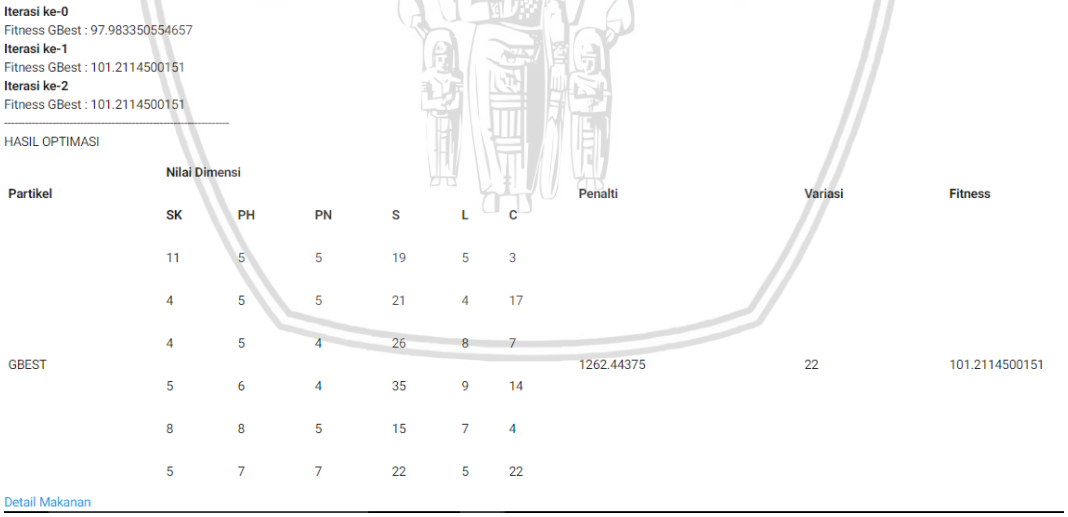
5.3.3 Implementasi Antarmuka Halaman *Output*

Halaman *output* ini keluar saat *user* selesai memasukkan data yang dibutuhkan. Pada halaman ini dibagi menjadi 2 bagian, bagian pertama menunjukkan proses perhitungan algoritme *Improved*-PSO dan bagian kedua adalah hasil *decode* dari optimasi yang dihasilkan oleh program yang menunjukkan

hasil akhir program. Implementasi antarmuka halaman *output* ditunjukkan pada Gambar 5.3, 5.4, 5.6 dan 5.7.

DATA DIRI
Nama : xxx
Tinggi Badan : 150 cm
Berat Badan : 50 Kg
Usia : 60 Tahun
Jenis Kelamin : Wanita
Pekerjaan : SBerat
BBI (Berat Badan Ideal) : 45 Kg
IMT (Indeks Massa Tubuh) : 22.222222222222
Kategori Berat Badan : Normal
Kebutuhan Kalori Basal Jenis Kelamin : 1125
Kebutuhan Kalori Basal Usia : 112.5
Kebutuhan Kalori Basal Aktivitas : 562.5
Kebutuhan Kalori Basal Berat Badan : 225
Total Kalori Harian : 1800 KKal
Kebutuhan Karbohidrat : 225 KKal
Kebutuhan Protein : 135 KKal
Kebutuhan Lemak : 40 KKal

Gambar 5.3 Antarmuka Halaman *Output* Data Diri Pasien dan Hasil Perhitungan Kebutuhan Gizi Pasien



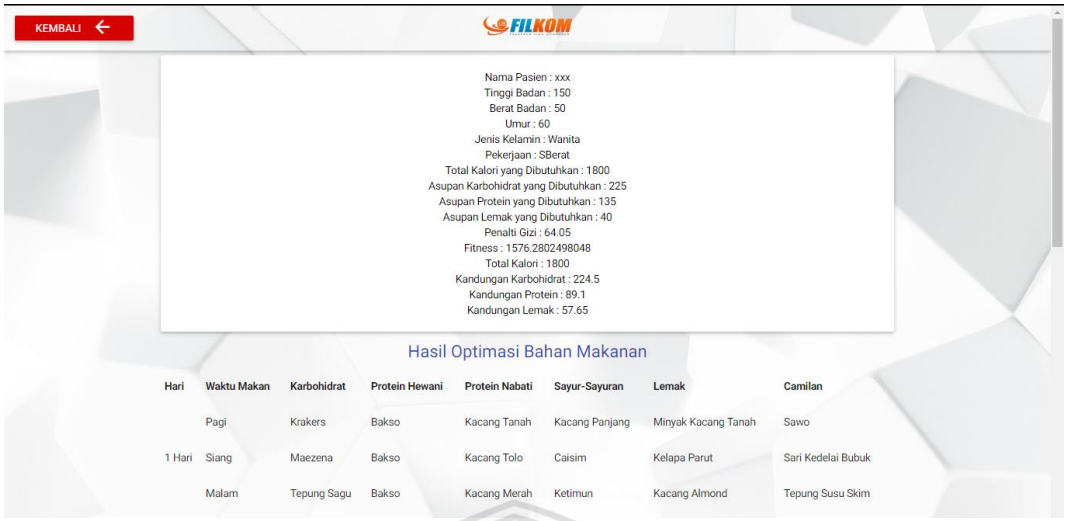
Iterasi ke-0
Fitness GBest : 97.983350554657
Iterasi ke-1
Fitness GBest : 101.2114500151
Iterasi ke-2
Fitness GBest : 101.2114500151

HASIL OPTIMASI

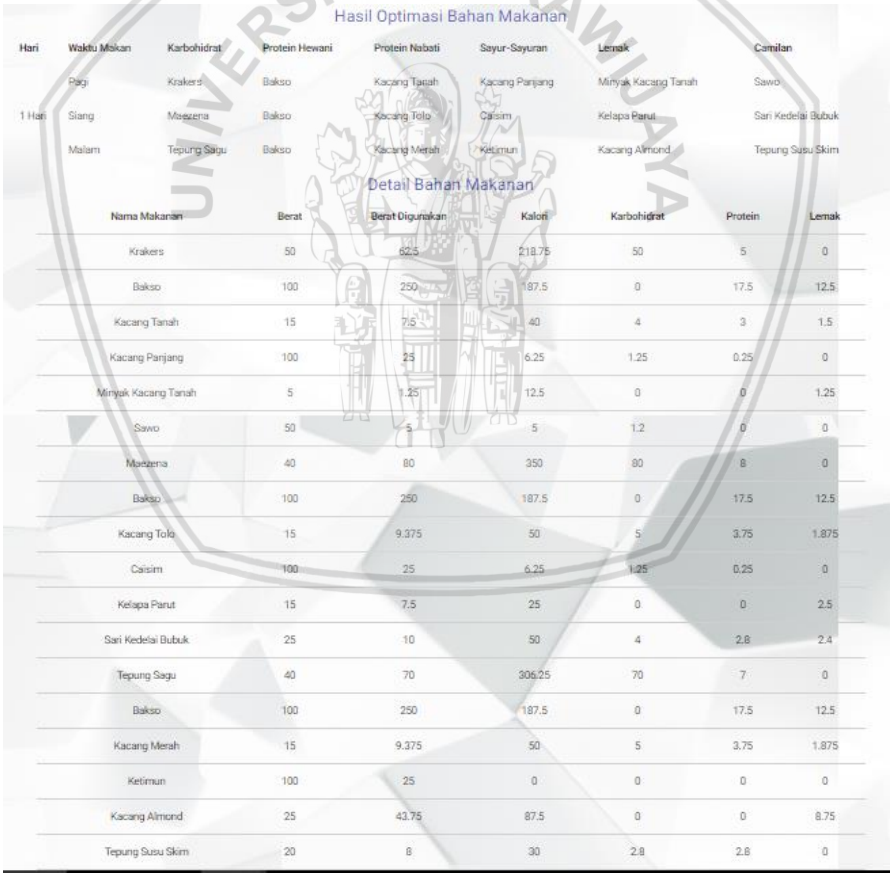
Partikel	Nilai Dimensi						Penalti	Variasi	Fitness
	SK	PH	PN	S	L	C			
	11	5	5	19	5	3			
	4	5	5	21	4	17			
	4	5	4	26	8	7			
GBEST							1262.44375	22	101.2114500151
	5	6	4	35	9	14			
	8	8	5	15	7	4			
	5	7	7	22	5	22			

Detail Makanan

Gambar 5.4 Antarmuka Halaman *Output* Perhitungan Tiap Iterasi



Gambar 5.5 Antarmuka Halaman *Output* Data Diri Pasien dan Hasil Rekomendasi Bahan Makanan Dari Sistem

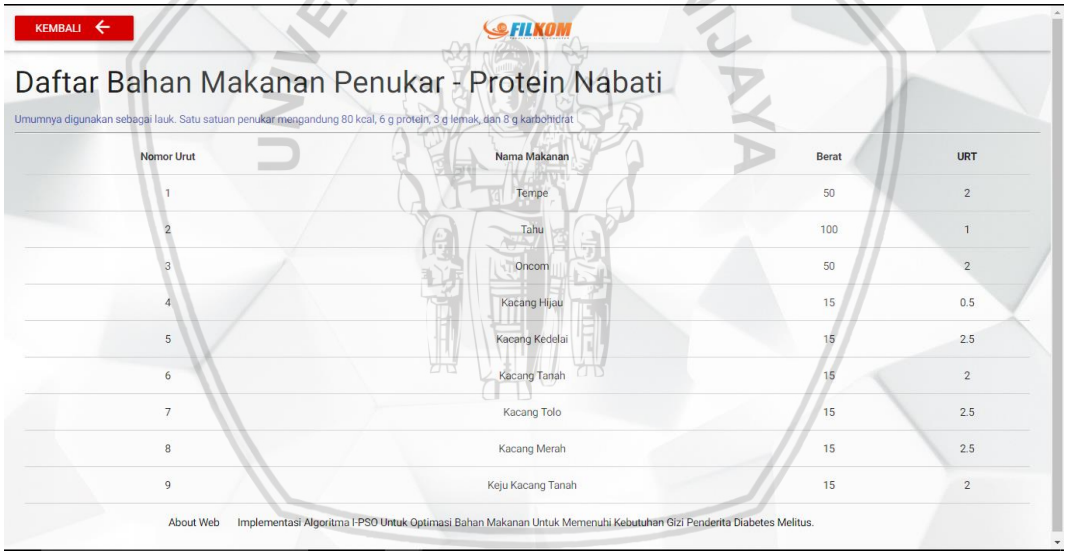


Gambar 5.6 Antarmuka Halaman *Output* Detail Bahan Makanan Hasil Rekomendasi

5.3.4 Implementasi Antarmuka Halaman Daftar Bahan Makanan Penukar

Pada halaman ini ditampilkan daftar bahan makanan penukar yang digunakan dalam perhitungan optimasi. Implementasi antarmuka halaman daftar bahan makanan penukar ditunjukkan pada Gambar 5.6 dan 5.7.

Gambar 5.7 Antarmuka Halaman Daftar Bahan Makanan Penukar (I)



Gambar 5.8 Antarmuka Halaman Daftar Bahan Makanan Penukar (II)

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini berisi mengenai pengujian dalam penelitian yang dilakukan serta pembahasan dari hasil pengujian tersebut. Pengujian ini dibagi dalam 3 macam pengujian seperti yang terdapat pada perancangan yang telah disusun. Pengujian pertama berupa pengujian parameter, kedua adalah pengujian konvergensi, dan yang terakhir merupakan analisis global.

6.1 Pengujian Parameter

Pengujian pertama yang dilakukan berupa pengujian parameter berupa jumlah populasi dan pengujian nilai C_1&C_2. Pengujian ini masing-masing dilakukan sebanyak 10 kali dengan kebutuhan gizi pasien sebesar 1732,5 kalori, kebutuhan karbohidrat 281,53125 gr, kebutuhan protein 64,96875 gr, dan kebutuhan lemak 38,5 gr.

6.1.1 Pengujian Jumlah Populasi

Pengujian jumlah populasi atau *popSize* dilakukan untuk mengetahui berapa jumlah populasi yang paling optimal yang bekerja pada sistem, sehingga sistem dapat menghasilkan solusi yang optimal dalam memberi komposisi bahan makanan untuk pemenuhan gizi penderita Diabetes Melitus. Parameter yang digunakan dalam pengujian ini adalah:

- Jumlah Iterasi : 10
- Jumlah Hari : 2
- C_1&C_2 : dinamis
- r_1&r_2 : *random*[0,1]

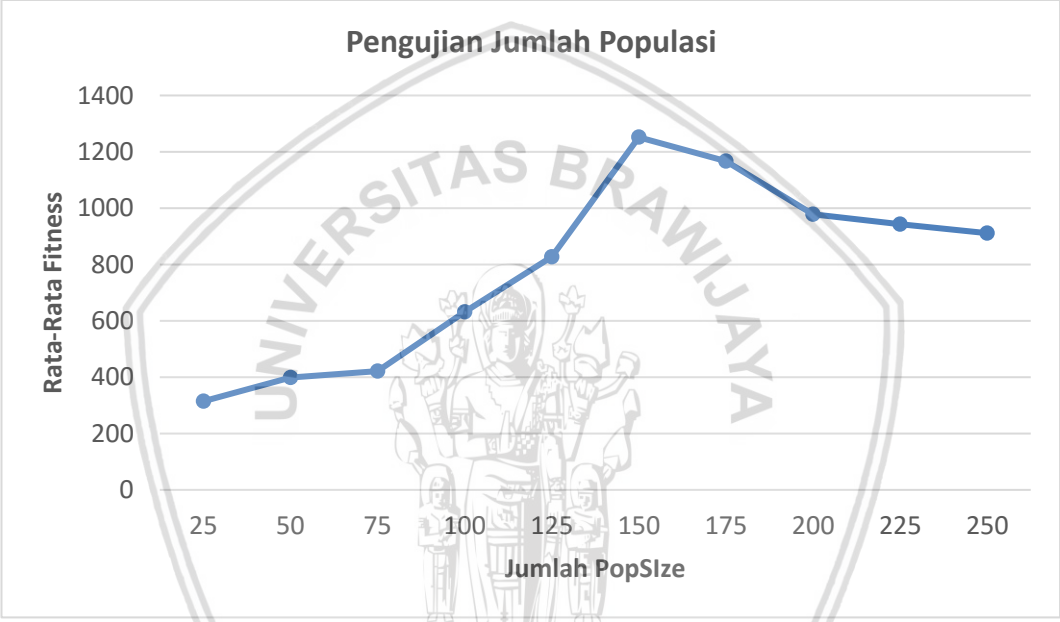
Hasil pengujian jumlah populasi ditunjukkan pada Tabel 6.1 dengan menunjukkan rata-rata nilai *fitness* yang dihasilkan dari beberapa jumlah populasi yang berbeda. Hasil yang lebih detail dapat dilihat pada Lampiran D.

Tabel 6.1 Hasil Pengujian Jumlah Populasi

Jumlah Populasi	Nilai *Fitness* Percobaan ke-*i*				Jumlah *Fitness*	Rata-Rata *Fitness*
	1	2	…	10		
25	177.876	254.767		216.4432	3144.2201	314.42201
50	499.1061	265.6846		316.4177	3987.4485	398.74485
75	343.9968	515.5967		596.1258	4216.7583	421.67583
100	682.4097	1891.536		344.687	6316.6607	631.66607
125	596.5602	330.5822		637.5938	8265.8039	826.58039
150	803.2845	777.5727		910.656	12515.0266	1251.50266

Jumlah Populasi	Nilai *Fitness* Percobaan ke-*i*				Jumlah *Fitness*	Rata-Rata *Fitness*
	1	2	...	10		
175	770.3092	495.5834		583.5579	11669.6989	1166.96989
200	610.4938	321.878		850.5743	9780.4298	978.04298
225	613.3375	631.3375		936.6427	9427.7835	942.77835
250	458.8242	496.1073		729.5742	9109.0951	910.90951

Berdasarkan pada Tabel 6.1, dibuatlah Grafik Hasil Pengujian Jumlah Populasi yang dapat dilihat pada Gambar 6.1.



Gambar 6.1 Grafik Hasil Pengujian Jumlah Populasi

Berdasarkan pada Grafik Hasil Pengujian Jumlah Populasi, terdapat kenaikan rata-rata nilai *fitness* di tiap kenaikan jumlah populasi meskipun terdapat penurunan di beberapa titik tertentu. Kenaikan nilai *fitness* ini merupakan dampak dari banyaknya jumlah partikel yang bisa memberikan solusi yang lebih bervariasi dan menimbulkan ruang pencarian solusi yang besar dan luas. Luasnya ruang pencarian solusi tersebut mempermudah kinerja sistem dalam memperoleh penyelesaian masalah yang paling optimal. Dari pengujian di atas, nilai rata-rata *fitness* terendah adalah pada titik 314,42201 dengan jumlah populasi 25 dan nilai rata-rata *fitness* tertinggi adalah pada titik 1251,50266 dengan jumlah populasi 150. Dari hasil di atas dapat disimpulkan bahwa jumlah populasi 150 merupakan jumlah yang paling optimal dalam mendapatkan solusi yang terbaik.

6.1.2 Pengujian Koefisien Akselerasi

Parameter C_1 dan C_2 yang kemudian disebut koefisien akselerasi berfungsi untuk menyeimbangkan antara global eksplorasi dan lokal eksploitasi agar

menjadi lebih baik. Global eksploitasi merupakan nilai *fitness* tertinggi yang didapatkan oleh seluruh partikel dalam suatu populasi, sedangkan lokal eksploitasi merupakan *fitness* tertinggi yang didapatkan oleh satu partikel saat itu juga. Nilai koefisien akselerasi yang besar mengakibatkan pergerakan partikel dalam menempati posisinya yang baru menjadi relatif lebih jauh dari pergerakan biasanya sehingga kemampuan menjelajah partikel tersebut menjadi lebih luas dan baik namun terkadang dapat melebihi atau kurang dari batas pencarian. Pada pengujian koefisien akselerasi ini dicari kombinasi nilai C_1 dan C_2 terbaik dalam pencarian solusi pada sistem. Parameter yang digunakan pada pengujian ini adalah:

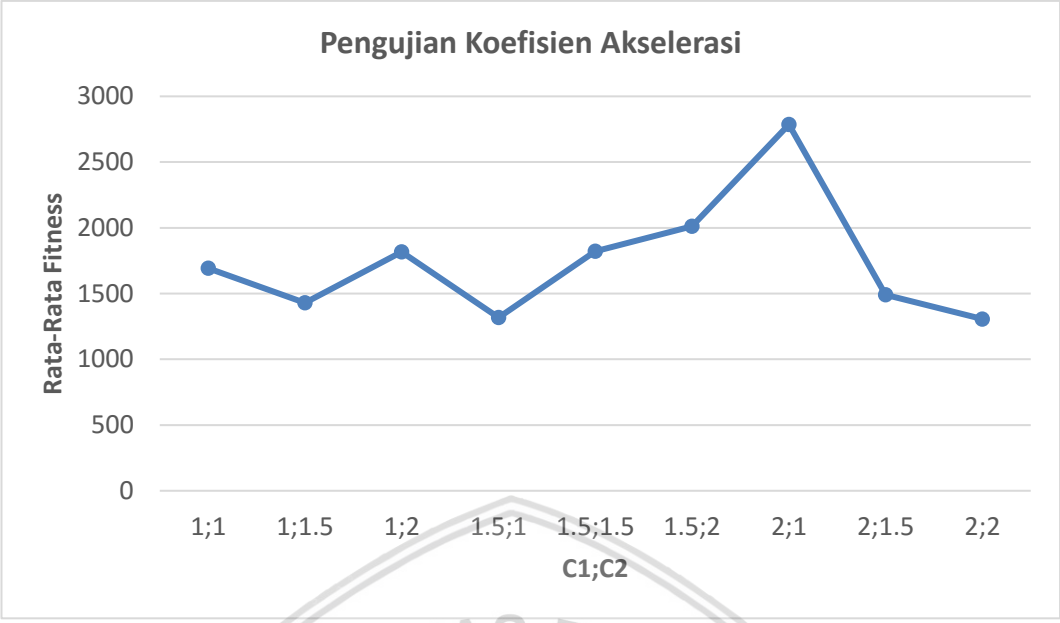
- Jumlah Populasi : 150
- Jumlah Iterasi : 10
- Jumlah Hari : 2
- r_1&r_2 : *random*[0,1]

Hasil pengujian koefisien akselerasi ditunjukkan pada Tabel 6.2 dengan menunjukkan rata-rata nilai *fitness* yang dihasilkan dari beberapa kombinasi nilai C_1 dan C_2 yang berbeda-beda. Hasil yang lebih mendetail dapat dilihat pada Lampiran D.

Tabel 6.2 Hasil Pengujian Koefisien Akselerasi

Koefisien Akselerasi		**Nilai *Fitness* Percobaan ke-*i***				**Jumlah *Fitness***	**Rata-Rata *Fitness***
C₁	***C₂***	**1**	**2**	**…**	**10**		
1	1	1059.974	402.6746		1407.854	16929.2053	1692.92053
	1.5	666.0286	518.473		2618.78	14295.1427	1429.51427
	2	2966.029	4704.1		318.7046	18164.5509	1816.45509
1.5	1	592.532	912.0501		1137.604	13163.4254	1316.34254
	1.5	2419.557	4121.41		3470.564	18213.2903	1821.32903
	2	372.895	739.3313		303.826	20105.7629	2010.57629
2	1	6803.411	949.6951		756.6863	27857.5863	2785.75863
	1.5	4899.025	1468.654		603.3383	14897.5372	1489.75372
	2	928.0797	1622.973		459.7472	13064.9763	1306.49763

Berdasarkan pada Tabel 6.2, dibuatlah Grafik Hasil Pengujian Jumlah Populasi yang dapat dilihat pada Gambar 6.2.



Gambar 6.2 Grafik Hasil Pengujian Koefisien Akselerasi

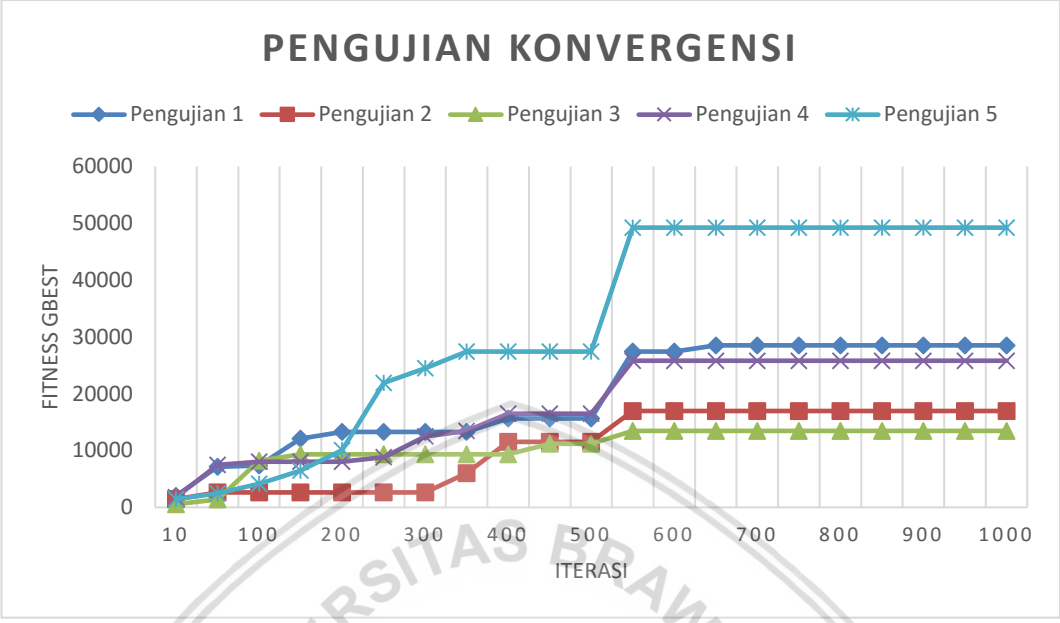
Koefisien akselerasi merupakan pemegang kendali dalam pergerakan partikel di ruang pencarian. Nilai koefisien akselerasi yang terlampau besar mengakibatkan pergerakan partikel dalam mencari posisi baru menjadi relatif lebih jauh. Namun, bila nilai koefisien akselerasi terlampau kecil juga mengakibatkan pergerakan partikel terjebak pada lokal optimum. Dari kedua kondisi di atas, maka perlu dibentuk sebuah kombinasi nilai koefisien akselerasi agar bisa membuat pergerakan partikel lebih optimal dalam tujuannya mencari posisi terbaik dan menghasilkan banyak variasi. Berdasarkan pada Grafik Hasil Pengujian Koefisien Akselerasi, kombinasi nilai C_1 dan C_2 yaitu sebesar 2 dan 1 adalah kombinasi terbaik dengan nilai rata-rata *fitness* tertinggi sebesar 2785,75863. Kesimpulannya adalah kombinasi tersebut merupakan kombinasi nilai koefisien akselerasi yang paling optimal.

6.2 Pengujian Konvergensi

Pengujian yang kedua adalah pengujian konvergensi yang berfungsi untuk mengetahui titik konvergen sistem terdapat pada iterasi tertentu. Pengujian konvergensi ini dilakukan sebanyak 5 kali dengan jumlah iterasi maksimal adalah 1000 iterasi. Untuk parameter algoritme yang digunakan memakai parameter hasil pengujian sebelumnya agar mendapatkan hasil yang paling optimal. Berikut rincian parameter yang digunakan:

- Jumlah Populasi : 150
- Jumlah Hari : 2
- Nilai C_1&C_2 : 2;1
- r_1&r_2 : *random*[0,1]

Detail hasil pengujian konvergensi yang mendetail dapat dilihat pada Lampiran D dan ditunjukkan dalam bentuk grafik seperti pada Gambar 6.3 berikut.



Gambar 6.3 Grafik Hasil Pengujian Konvergensi

Berdasarkan pada grafik hasil pengujian di atas dapat dilihat bahwa dengan menggunakan 1000 iterasi, nilai *fitness GBest* yang dihasilkan selalu mengalami peningkatan seiring dengan meningkatnya jumlah iterasi. Dari keseluruhan percobaan, nilai *fitness* mencapai hasil optimal dan menyentuh titik konvergen pada iterasi ke-550 meskipun terdapat satu titik pada iterasi ke-650. Nilai *fitness* yang dihasilkan dari awal iterasi memiliki nilai yang cukup rendah, namun selalu mengalami perbaikan dan mencapai titik konvergen pada iterasi ke-550. Konvergen adalah ketika keragaman populasi semakin berkurang yang diakibatkan oleh proses pembaharuan yang terus menerus dan selisih nilai *fitness* yang dihasilkan dari iterasi satu ke yang lainnya memiliki selisih sejumlah 0.

6.3 Analisis Global

Analisis global menggunakan parameter-parameter terbaik yang sebelumnya didapatkan dari pengujian sebelumnya. Parameter yang digunakan adalah:

- Jumlah Populasi : 150
- Jumlah Hari : 1
- Nilai C_1&C_2 : 2;1
- r_1&r_2 : *random*[0,1]

Analisis global ini dilakukan dengan cara menggunakan parameter optimal di atas yang digunakan untuk menguji sistem dengan membandingkan hasilnya dengan data sampel perhitungan kebutuhan gizi aktual yang didapatkan dari pakar. Data pasien yang diambil adalah 10 data pasien dengan 2 jenis kualifikasi, yaitu kualifikasi berdasarkan kategori aktivitas dan kondisi berat badan. Berikut

adalah daftar pasien lengkap dengan kebutuhan gizi serta data pribadi masing-masing yang dapat dilihat pada Tabel 6.3.

Tabel 6.3 Data Perhitungan Gizi Manual

NO.	Kualifikasi	JK	Usia	BB	TB	Kal	KH	P	L
Pasien 1	Bedrest	P	40	50	150	1250.0	203	46	27.8
Pasien 2	Ringan	L	45	57	160	1966.5	319	73	43.7
Pasien 3	Sedang	P	50	50	150	1562.2	253	58	34.0
Pasien 4	Berat	L	55	57	160	2308.5	375	86	51.3
Pasien 5	Sangat Berat	P	60	50	150	1750.0	284	65	38.0
Pasien 6	Kurus	L	65	45	160	1552.5	252	58	34.5
Pasien 7	Normal	P	70	50	150	1250.0	203	46	27.0
Pasien 8	Overweight	L	40	63	160	2173.5	353	81	48.3
Pasien 9	Obese I	P	45	68	150	1615	262	60	35.9
Pasien 10	Obese II	L	50	77	160	2425.5	394	91	53.9

(Sumber: Fauziatul Firdaus, S.Gz, Ahli Gizi Diet Indo)

Langkah berikutnya dalam analisis global adalah melakukan pengujian sistem dengan menggunakan parameter terbaik sekaligus dengan data aktual pasien sebagai perhitungannya. Berikut hasil perhitungan gizi yang dihasilkan oleh sistem dapat dilihat pada Tabel 6.4.

Tabel 6.4 Data Perhitungan Gizi Hasil Rekomendasi Sistem

NO.	Kualifikasi	JK	Usia	BB	TB	Kal	KH	P	L
Pasien 1	Bedrest	P	40	50	150	1250.0	202.2	47.25	27.0
Pasien 2	Ringan	L	45	57	160	1967.5	295.469	75.419	51.2
Pasien 3	Sedang	P	50	50	150	1561.9	253.6	58.65	34.0
Pasien 4	Berat	L	55	57	160	2017.5	300	78.35	53.3
Pasien 5	Sangat Berat	P	60	50	150	1750.6	285.75	73.05	34.5
Pasien 6	Kurus	L	65	45	160	1417.5	230.45	51.85	30.9
Pasien 7	Normal	P	70	50	150	1248.8	196.6	48.7	29.0
Pasien 8	Overweight	L	40	63	160	2161.9	299.75	91.3	62.4
Pasien 9	Obese I	P	45	68	150	1445.6	234	54.2	31.5
Pasien 10	Obese II	L	50	77	160	1961.9	297.25	75.05	49.3

Berikut adalah Tabel 6.5 yang menunjukkan perhitungan selisih antara data kebutuhan gizi aktual dengan kebutuhan gizi dari sistem.

Tabel 6.5 Selisih Kandungan Gizi

No	Nama	Selisih Kalori	Selisih Karbohidrat	Selisih Protein	Selisih Lemak
1	Pasien 1	0%	0.39%	-2.71%	2.81%
2	Pasien 2	-0.05%	7.38%	-3.31%	-17.22%
3	Pasien 3	0.02%	-0.23%	-1.12%	-0.07%
4	Pasien 4	12.61%	20%	8.9%	-3.9%
5	Pasien 5	-0.04%	-0.62%	-12.39%	9.28%
6	Pasien 6	8.7%	8.55%	10.6%	10.51%
7	Pasien 7	0.1%	3.15%	-5.87%	-7.41%
8	Pasien 8	0.53%	15.08%	-12.72%	-29.14%
9	Pasien 9	10.49%	10.69%	9.67%	12.26%
10	Pasien 10	19.11%	24.56%	17.53%	8.63%
Rata-Rata Selisih		5.15%	8.9%	0.86%	-1.43%

Berdasarkan pada Tabel 6.5 di atas, dihasilkan rata-rata selisih kebutuhan kalori, karbohidrat, protein, dan lemak aktual dengan hasil rekomendasi sistem. Sebelumnya, dari hasil wawancara dengan Ahli Gizi, diketahui bahwa terdapat batas toleransi selisih kandungan gizi bagi pasien yaitu ±10%. Rata-rata selisih nilai kebutuhan kalori, karbohidrat, protein, dan lemak yang dihasilkan sistem masing-masing sebesar 5,15%, 8,9%, 0,86%, dan -1,43%.

Berdasarkan hasil selisih tersebut, kebutuhan kalori yang dihasilkan sistem mampu memenuhi gizi sebesar 94,85% yang berarti terdapat kekurangan 5,15% dari gizi manual. Untuk kebutuhan karbohidrat yang dihasilkan sistem mampu memenuhi gizi sebesar 91,1% yang berarti terdapat kekurangan 8,9% dari gizi manual. Untuk kebutuhan protein yang dihasilkan sistem mampu memenuhi gizi sebesar 99,14% yang berarti terdapat kekurangan hanya 0,86% dari gizi manual. Untuk kebutuhan lemak yang dihasilkan oleh sistem melebihi gizi manual sebesar 1,43%. Dengan hasil tersebut, sistem dianggap mampu memenuhi kebutuhan gizi pasien karena masih dalam batas toleransi Ahli Gizi, yaitu ±10%.

Setelah menghitung kebutuhan gizi penderita Diabetes Melitus yang dihasilkan oleh sistem, selanjutnya sistem akan menampilkan hasil dari rekomendasi komposisi bahan makanan yang optimal sesuai dengan kebutuhan gizi pasien. Berikut merupakan hasil rekomendasi komposisi bahan makanan hasil optimasi bagi pasien ditunjukkan pada Tabel 6.6 dan untuk selengkapnya dapat dilihat pada Lampiran E.

Tabel 6.6 Hasil Rekomendasi Komposisi Bahan Makanan

Hari Ke-	Waktu Makan	Nama Bahan Makanan	Berat Asli (g)	Total Berat (g)	Kalori (kcal)	Karbohidrat (g)	Protein (g)	Lemak (g)
1	Pagi	Tepung Sagu	40	70	306.25	70	7	0
		Bakso	100	250	187.5	0	17.5	12.5
		Oncom	50	25	40	4	3	1.5
		Pepaya Muda	100	25	6.25	1.25	0.25	0
		Kelapa Parut	15	7.5	25	0	0	2.5
	Camilan	Sirsak	50	2.5	2.5	0.6	0	0
	Siang	Tepung Terigu	50	100	350	80	8	0
		Udang Segar	35	8.75	18.75	0	1.75	1.25
		Keju Kacang Tanah	15	7.5	40	4	3	1.5
		Brokoli	100	25	6.25	1.25	0.25	0
		Kelapa Parut	15	7.5	25	0	0	2.5
	Camilan	Pear	75	3.75	2.5	0.6	0	0
	Malam	Mie Kering	50	12.5	43.75	10	1	0
		Telur Ayam	50	12.5	18.75	0	1.75	1.25
		Kacang Tanah	15	7.5	40	4	3	1.5
		Daun Talas	100	25	12.5	2.5	0.75	0
		Kelapa Parut	15	7.5	25	0	0	2.5
	Camilan	Anggur	165	330	100	24	0	0

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan penelitian mengenai optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita Diabetes Melitus, didapatkan kesimpulan sebagai berikut:

1. Algoritme *Improved Particle Swarm Optimization* (IPSO) dapat diimplementasikan untuk optimasi komposisi bahan makanan untuk memenuhi kebutuhan gizi penderita diabetes meltius. Hasil akhir yang dimunculkan berupa rekomendasi komposisi bahan makanan yang dapat dikonsumsi oleh pasien selama maksimal 3 hari dengan waktu makan 3 kali sehari. Pasien diabetes dibatasi dengan rentang usia di atas 40 tahun dengan kondisi diabetes tanpa komplikasi. Inisialisasi partikel menggunakan *Real-Code PSO* (RCPSO) untuk memaksimalkan nilai partikel. Terdapat 2 parameter tambahan sebagai pengembangan algoritme PSO menjadi IPSO yaitu *constriction factor* dan *inertia weight* untuk meningkatkan optimasi yang dihasilkan oleh algoritme IPSO. Setelah mengimplementasi algoritme *Improved*-PSO, parameter optimal yang didapatkan adalah jumlah populasi sebesar 150, nilai koefisien akselerasi 2;1, dan sistem mencapai konvergen pada iterasi ke-550. Parameter optimal inilah yang kemudian digunakan dalam penyelesaian masalah optimasi komposisi bahan makanan.
2. Berdasarkan parameter optimal tersebut sistem dapat menghasilkan rata-rata selisih kalori, karbohidrat, protein, dan lemak sebesar 5,15%, 8,9%, 0,86%, dan -1,43. Dari hasil evaluasi tersebut dapat disimpulkan bahwa sistem mampu menghasilkan komposisi bahan makanan untuk memenuhi kebutuhan gizi pasien yang optimal karena masih memenuhi batas toleransi dari Ahli Gizi, yaitu sebesar ±10%.

7.2 Saran

Sebagai pengembangan pada penelitian selanjutnya, terdapat beberapa saran yang dapat digunakan untuk memperbaiki hasil penelitian selanjutnya:

1. Pada penelitian selanjutnya dapat ditambahkan parameter harga bahan makanan agar dapat menghasilkan komposisi bahan makanan dengan harga yang lebih terjangkau. Serta terdapat rekap belanja untuk membantu pasien mendapatkan bahan makanan yang mudah dibeli dengan tidak melupakan variasi bahan makanan.
2. Data pasien untuk penelitian selanjutnya dapat dikembangkan dengan memberikan hasil rekomendasi komposisi bahan makanan dalam jangka waktu yang lebih panjang dan terdapat pembaruan menu bahan makanan sesuai anjuran Ahli Gizi.

DAFTAR PUSTAKA

Cholissodin, Imam. dan Riyandani, Efi., 2016. Swarm Intellligence (Teori & Case Study).

Eliantara, Felia., Cholissodin, Imam., Indriati., 2016. Optimasi Pemenuhan Kebutuhan Gizi Keluarga Menggunakan Particle Swarm Optimization.

Harahap, Syaiful W., 2016. Mencegah Diabetes di Hulu. [online] Tersedia di: <https://www.kompasiana.com/infokespro/mencegah-diabetes-di-hulu_58182b681697730816bae20b> [Diakses 29 Januari 2018].

Hasjidla, Nur Firra., Cholissodin, Imam., Widodo, Agus Wahyu., 2018. Optimasi Komposisi Pakan Untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum Menggunakan *Improved* Particle Swarm Optimization (*IMPROVED*-PSO).

HL Chen, et all, 2011. An Adaptive Fuzzy K-Nearest Neighbor Method Based on Parallel Particle Swarm Optimization for Bankcruptcy Prediction, Part 1.

International Diabetes Federation, 2017. IDF Western Pacific members. [online] Tersedia di: <https://www.idf.org/our-network/regions-members/western-pacific/members/104-indonesia.html> [Diakses 29 Januari 2018].

Istikomah, Leni., Cholissodin, Imam., Marji., 2017. Implementasi Algoritme Particle Swarm Optimization (PSO) untuk Optimasi Pemenuhan Kebutuhan Gizi Balita.

Kementrian Kesehatan Republik Indonesia, 2016. Menkes: Mari Kita Cegah Diabetes Dengan Cerdik. [online] Tersedia di <http://www.depkes.go.id/article/print/16040700002/menkes-mari-kita-cegah-diabetes-dengan-cerdik.html> [Diakses 15 Maret 2018]

Marianti, dr., 2016. Pengertian Diabetes. [online] Tersedia di: <http://www.alodokter.com/diabetes> [Diakses 29 Januari 2018].

Maryamah, Putri., Rekyan Regasari Mardi., Wicaksono, Satrio Agung., 2017. Optimasi Komposisi Makanan Pada Penderita Diabetes Melitus dan Komplikasinya Menggunakan Algoritme Genetika.

Misdarina., Ariani, Yesi., 2012. Pengetahuan Diabetes Melitus dengan Kadar Gula Darah pada Pasien DM Tipe 2.

Ratnaweera A., Halgamuge, SK., Watson HC., 2004. Self Organizing Hierarchical Particle Swarm Optimizer with Time-Varying Acceleration Coefficients.

Simamora, Veronica Kristina BR., Cholissodin, Imam., Fauzi, M Ali., 2017. Optimasi Biaya Bahan Menu Makanan bagi Penderita Penyakit Jantung dengan Menggunakan Metode Evolution Strategies.

Soelistijo, Soebagijo Adi., Novida, Hermina., Rudijanto, Achmad., et. Al. 2015. Pengelolaan dan Pencegahan Diabetes Melitus Tipe 2 di Indonesia.

Suryani, Nany., Pramono., Septiana, Henny., 2015. Diet dan Olahraga sebagai Upaya Pengendalian Kadar Gula Darah pada Pasien Diabetes Melitus Tipe 2 di Poliklinik Penyakit Dalam RSUD Ulin Banjarmasin.

Trijayanto, Puput Aji., 2016. Hubungan Riwayat Garis Keturunan Dengan Waktu Terdiagnosis Diabetes Melitus di RSUD. Prof. DR. Margono Soekarjo Purwokerto.

Yonghe, L., Minghui, L., Zeyuan, Y. & Lichao, C., 2015. *Improved* Particle Swarm Optimization Algorithm and Its Application in Text Feature Selection.